# ESCUELA POLITÉCNICA NACIONAL

## ESCUELA DE FORMACIÓN DE TECNÓLOGOS

## IMPLEMENTACIÓN DE PROTOTIPOS PARA TRANSMISIÓN INALÁMBRICA BASADOS EN TECNOLOGÍA LORA

## IMPLEMENTACIÓN DE UN PROTOTIPO PARA TRANSMISIÓN INALÁMBRICA DE TEMPERATURA Y HUMEDAD BASADO EN TECNOLOGÍA LORA.

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN REDES Y TELECOMUNICACIONES**

**OLMEDO LEMACHE MILENA SALOMÉ**

**DIRECTOR: ANDRÉS FERNANDO REYES CASTRO**

**DMQ, marzo 2022**

# CERTIFICACIONES

Yo, MILENA SALOMÉ OLMEDO LEMACHE declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.


**MILENA SALOMÉ OLMEDO LEMACHE**

**milena.olmedo@epn.edu.ec**

**milena.olmedo.25@gmail.com**


Certifico que el presente trabajo de integración curricular fue desarrollado por MILENA SALOMÉ OLMEDO LEMACHE, bajo mi supervisión.


**ANDRÉS FERNANDO REYES CASTRO**

**DIRECTOR**

**andres.reyes@epn.edu.ec**

# DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

**MILENA SALOME OLMEDO LEMACHE**

**milena.olmedo@epn.edu.ec**

**milena.olmedo.25@gmail.com**

# DEDICATORIA

Dedico este proyecto a mis padres Fanny y Carlos, que forjaron mi carácter y me apoyaron incondicionalmente en mis estudios. Gracias por cuidarme, por procurar mi salud y por ayudarme siempre que podían, este proyecto en parte es suyo. Todo lo que logré y lograré es gracias a ustedes.

Milena.

# AGRADECIMIENTO

En primer lugar, quiero agradecer a la comunidad politécnica, por darme la oportunidad de formarme académicamente. Al ingeniero Leandro Pazmiño, por desarrollar mi amor por la carrera, por sus numerosos consejos y por siempre procurar el bienestar de sus estudiantes.

Agradezco a mis hermanos Valeria y Mateo, por siempre subirme el ánimo, ser mi soporte emocional y compartir tiempo de calidad conmigo. A mis amigos, por escucharme en mis momentos más difíciles, por alegrarme mis días y por ayudarme cuando lo necesitaba. Gracias especiales a Leyla, por ser una amiga incondicional en mi vida; y a Eliam, por ser el mejor amigo que pude tener en la carrera.

Milena.

# ÍNDICE DE CONTENIDOS

CERTIFICACIONES ..... I

DECLARACIÓN DE AUTORÍA ..... II

DEDICATORIA ..... III

AGRADECIMIENTO ..... IV

ÍNDICE DE CONTENIDO ..... V

RESUMEN ..... VII

*ABSTRACT* ..... VIII

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO ..... 1

1.1 Objetivo general ..... 2

1.2 Objetivos específicos ..... 2

1.3 Alcance ..... 2

1.4 Marco Teórico ..... 3

Tecnología LoRa ..... 3

LoRa e IOT ..... 4

Heltec LoRa 32 ..... 4

Sensor DHT22 ..... 5

2 METODOLOGÍA ..... 6

3 RESULTADOS ..... 8

3.1 Requerimientos del prototipo ..... 8

3.2 Definición de los componentes de *hardware* y *software* ..... 9

Hardware ..... 9

Software ..... 11

3.3 Diseño del prototipo ..... 11

Circuito electrónico ..... 11

Consumo energético ..... 12

Diseño 3D del prototipo ..... 13

Diagrama de flujo ..... 14

3.4 Implementación del prototipo ..... 17
Elaboración de la placa electrónica ..... 17
Elaboración de las cajas de protección ..... 19
Programación de los módulos WiFi LoRa 32 ..... 21
Montaje del prototipo en las cajas de protección ..... 26
3.5 Pruebas de funcionamiento ..... 27
Resultados medición de temperatura y humedad ..... 27
Resultados de transmisión de datos ..... 31
4 CONCLUSIONES ..... 36
5 RECOMENDACIONES ..... 37
6 REFERENCIAS BIBLIOGRÁFICAS ..... 38
7 ANEXOS ..... 40
ANEXO I: Certificado de Originalidad ..... i
ANEXO II: Enlaces ..... ii
ANEXO III: Códigos Fuente ..... iii

# RESUMEN

El presente Trabajo de Integración Curricular expone la implementación de un prototipo portátil orientado a la transmisión inalámbrica de datos de humedad y temperatura. Esto se logra a través de la programación de un maestro y un esclavo, y su comunicación se realiza mediante el empleo de la tecnología de transmisión de larga distancia *Long Range* (LoRa), haciendo uso de antenas incorporadas en módulos LoRa.

La lectura de datos se realiza utilizando un sensor de humedad y temperatura, y para la detección de los mismos se emplea un microcontrolador con un módulo LoRa integrado; la unión de estas dos partes conforma al dispositivo transmisor, que tiene la capacidad de desplegar en una pantalla la información que se va a transmitir.

En cuanto al dispositivo receptor, este se encuentra conformado por únicamente un microcontrolador con un módulo LoRa y una pantalla integrados. Este dispositivo cumple con el rol de receptor y se encarga de desplegar las lecturas de humedad y temperatura recibidas, con el objetivo de contrastar dicha información con la enviada.

Finalmente, se llevan a cabo pruebas de funcionamiento para comprobar la validez de los datos recolectados por el módulo transmisor, efectuando una comparación entre dicha información y las lecturas obtenidas con un dispositivo externo de alta precisión. También se valida la correcta transmisión de la información desde módulo transmisor a módulo receptor, realizando una comparativa entre los datos desplegados en cada pantalla.

**PALABRAS CLAVE:** sensor de temperatura y humedad, módulo LoRa, transmisión inalámbrica.

# *ABSTRACT*

*This project presents the implementation of a portable prototype oriented to the wireless transmission of humidity and temperature data. This is achieved through the programming of a master and a slave, and its communication is done by using LoRa transmission technology, by using antennas incorporated in LoRa modules.*

*Data reading is performed using a humidity and temperature sensor, and for data detection a microcontroller with an integrated LoRa module is used; the union of these two parts makes up the transmitter device, which has the capacity to display the information to be transmitted on a screen.*

*As for the receiver device, it is made up of only a microcontroller with an integrated LoRa module and a display. This device plays the role of receiver and is the responsible for displaying the humidity and temperature readings received, to contrast this information with the information sent.*

*Finally, functional tests are carried out to check the validity of the data collected by the transmitter module, comparing this information with the readings obtained with an external high-precision device. The correct transmission of the information from the transmitter module to the receiver module is also validated, making a comparison between the data displayed on each screen.*



***KEY WORDS****: temperature and humidity sensor, LoRa module, wireless transmission.*

# 1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

El riego incorrecto en jardines y huertos, el riesgo de inundación y fugas de agua, cero controles de los parámetros de aire en una edificación, entre otros, son algunos problemas asociados a la falta de control de temperatura y humedad. Una solución para estos inconvenientes se presenta al poner en práctica el uso de sensores de temperatura y humedad, cuya función principal es la medición, con el fin de prevenir desastres futuros relacionados al porcentaje de agua en el ambiente y su temperatura. De esta manera será posible prolongar la vida útil de equipos electrónicos al realizar un control de los parámetros descritos anteriormente.

En el presente proyecto se realiza la implementación de un prototipo de sensor orientado a la transmisión y recepción de datos de temperatura y de humedad. Para realizar dicha implementación se utiliza el módulo Heltec WIFI LoRa32 (V2) que, junto al sensor de temperatura y humedad DHT22, funciona como un aparato transmisor encargado de desplegar los datos recolectados con dicho sensor y enviarlos mediante transmisión inalámbrica a frecuencia de 915 MHz. La tecnología inalámbrica utilizada en el prototipo es *Long Range* (LoRa), cuyas ventajas son un bajo consumo de energía, largo alcance y conectividad segura.

La información es enviada de manera inalámbrica a un segundo módulo, que funciona como receptor para desplegar los datos de temperatura y humedad en la pantalla oled incorporada en el mismo. En la Figura 1.1 se presenta de manera sencilla el principio de funcionamiento del prototipo.

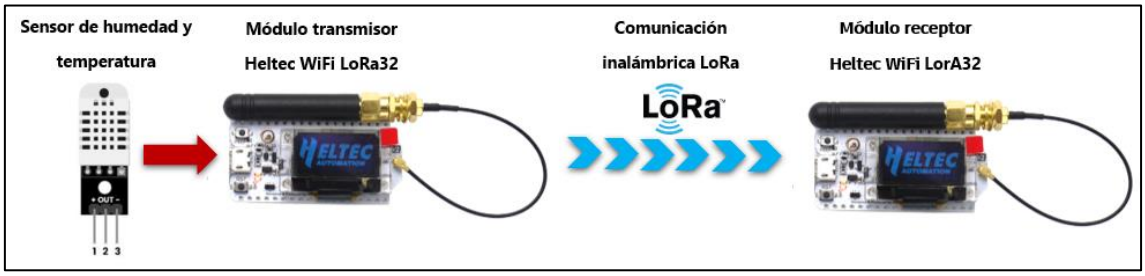


**Figura 1.1** Esquema de funcionamiento del prototipo [1] [2] [3]

Cada módulo está implementado para que sea portable, es decir, su alimentación se da utilizando una batería. Con esto se provee portabilidad al prototipo, lo que le permite ser utilizado en gran variedad de ambientes como, por ejemplo, invernaderos, zonas montañosas, zonas calidad, lugares húmedos, entre otros.

## 1.1 Objetivo general

Implementar un prototipo para transmisión inalámbrica de temperatura y humedad basado en tecnología LoRa.

## 1.2 Objetivos específicos

- Establecer los requerimientos del prototipo.
- Definir los componentes de *hardware* y *software*.
- Diseñar el prototipo.
- Implementar el prototipo.
- Realizar pruebas de funcionamiento.

## 1.3 Alcance

Mediante un análisis de los requerimientos de funcionalidad del prototipo, se determinan sus necesidades. Dentro de este estudio, se incluye una comparativa y selección de los dispositivos más adecuados para el prototipo, entre ellos se encuentran: módulo LoRa con pantalla oled incorporada, sensor de temperatura y humedad relativa, y batería para la alimentación.

En cuanto al prototipo, se incluye un diseño en 3D del mismo haciendo uso de un *software* especializado que provea una vista previa de su implementación. Además, se incluyen los análisis necesarios para saber cómo proporcionar la alimentación de las placas y permitir la portabilidad, seguido de su respectivo diagrama de flujo que describe detalladamente el código para lectura y transmisión de datos con LoRa.

En su implementación se tiene en cuenta la elaboración de una PCB (*Printed Circuit Board*) para el transmisor (Tx); de la misma manera, se construyen cajas de protección, capaces de proporcionar una buena estabilidad, durabilidad y estética tanto para el transmisor como para el receptor.

En las pruebas de funcionamiento, se realiza una comparación entre los datos transmitidos y recibidos de manera inalámbrica (temperatura y humedad), mediante tecnología LoRa. De esta manera, se verifica la congruencia entre ambos, anticipando pequeños errores en los valores debido a factores justificables; por ejemplo, la distancia y la presencia de obstáculos.

## 1.4 Marco Teórico

### Tecnología LoRa

Administrada por *LoRa Alliance*, es una tecnología inalámbrica que emplea la modulación CSS, cuyas siglas significan *Chirp Spread Spectrum*, patentada por Semtech. Esta tecnología es ideal para realizar conexiones a distancias largas y para implementar en redes de Internet de las cosas (IoT, *Internet of things*), en las que los sensores tengan problemas con el acceso a corriente eléctrica de red. Esta tecnología se puede aplicar en ciudades inteligentes, lugares con cobertura corta (zonas rurales) y redes privadas de sensores y/o actuadores [4].

El nombre LoRa hace alusión al largo alcance que tiene (10 a 20 km). LoRa presenta una tolerancia bastante alta a interferencias, al igual que una gran sensibilidad en la recepción de datos (alrededor de -139dB) y un consumo de batería bajo, pues pueden llegar a durar incluso 10 años con una sola batería. Su tasa de transferencia de datos es baja, presentando un máximo de 255 bytes, lo que puede parecer bajo, pero en realidad la vuelve ideal para la transmisión de datos recolectados en lecturas de sensores, ya que su peso es bastante pequeño [4].

Los dispositivos LoRa trabajan con una conexión punto a punto y las frecuencias que maneja dependen de la región de trabajo, como se muestra en la Tabla 1.1 Frecuencias de trabajo de LoRa [4]:

**Tabla 1.1** Frecuencias de trabajo de LoRa [4]

| Región | Frecuencia |
|---|---|
| Europa | 868 (MHz) |
| América | 915 (MHz) |
| Asia | 433 (MHz) |

Para el caso del módulo con el que se va a trabajar, la frecuencia utilizada es la de 915Mhz. Hay que tomar en cuenta que esta banda en Ecuador es de uso libre, y que el desarrollo del proyecto forma parte de aplicaciones Industriales, Científicas y Médicas (ICM). Estas bandas se encuentran definidas dentro del Reglamento de Radiocomunicaciones de la UIT, siendo la banda de frecuencia manejada de 902-928 (MHz) y su frecuencia central de 915 (MHz) [5].

### LoRa e IOT

LoRa es una tecnología de comunicación relativamente nueva bajo la Red de Área Amplia de Baja Potencia (LPWAN, *Low Power Wide Area Network*). Ya que hace hincapié en la comunicación de largo alcance con alta capacidad y sensibilidad de recepción, es capaz de trabajar con interferencia de ruido o piso de ruido de manera efectiva. La técnica de modulación utilizada en LoRa la vuelve resistente al ruido de canal, pues todo el ancho de banda asignado es utilizado para transmitir una señal de información o de datos. Además, la seguridad del sistema LoRa se puede garantizar ya que la transmisión se propaga de forma pseudoaleatoria presentada como un ruido, por lo que dicha técnica de modulación es la que proporciona la seguridad básica para el sistema LoRa. Por estas razones, LoRa es la mejor opción de implementación en soluciones IoT que necesitan un largo alcance y bajo uso de energía. En otras palabras, esta tecnología provee máxima eficiencia en comunicación mientras mantiene un bajo costo de desarrollo [6].

Una de las aplicaciones más típicas e importantes de redes LoRa es el monitoreo ambiental. En implementaciones de monitoreo ambiental, donde son necesarios los indicadores ambientales en tiempo real como la calidad del aire, la temperatura y la humedad, y teniendo en cuenta los requisitos financieros y energéticos, la implementación de una red con tecnología LoRa es adecuada para grandes áreas geográficas. De esta manera los nodos LoRa (de baja potencia) pueden interactuar con el entorno y reportar datos oportunamente para detectar cualquier problema antes de que se convierta en crisis. [7].

### Heltec LoRa 32

El microcontrolador WiFi LoRa 32 (V2) es una placa de desarrollo IoT diseñada y producida por la marca Heltec Automation. Se basa en ESP32+SX127x y presenta funciones Wi-Fi, Bluetooth y LoRa. Su diseño también incluye un sistema de gestión de batería Li-Po y una pantalla OLED de 2.44 (cm) [matriz de 128x64 puntos], como se muestra en la Figura 1.2 [8].

El chip LoRa con el que trabaja puede ser SX1276 o SX1278, dependiendo de la frecuencia en la que se desee manejar el dispositivo; en el caso del presente proyecto se trabaja con el SX1276. Su máximo poder de transmisión es de 19 (dB) ± 1 (dB), y su sensibilidad de recepción es de -135 (dBm) [8].

Ya que soporta el entorno de desarrollo Arduino (Arduino IDE), es un módulo fácil de programar, volviéndolo una opción atractiva para implementar en ciudades y granjas inteligentes, donde la medición de parámetros ambientales es necesaria.

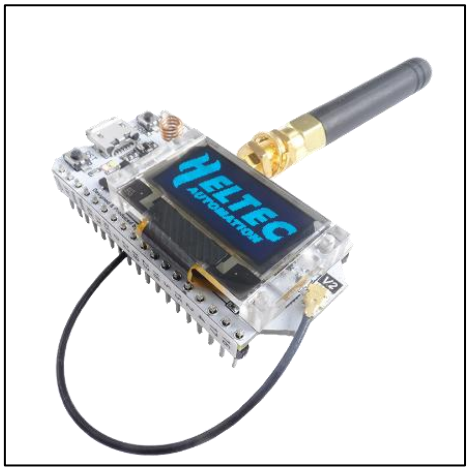


**Figura 1.2** Módulo Heltec Wi-Fi Lora32 [8]

### Sensor DHT22

El DHT22 es un sensor digital que permite monitorear tanto la temperatura como la humedad relativa. Se puede conseguir como sensor (4 pines) o módulo (3 pines), como se muestra en la Figura 1.3, ambos presentando un buen rendimiento y bajo costo. El sensor tiene un rango de medición de temperatura de -40 (°C) a 80 (°C), y de humedad de 0% hasta 100%, con una precisión de ±1°C y ±1%, respectivamente [9].

El uso de este sensor con Arduino es muy sencillo, pues a nivel de *software* existe disponibilidad de librerías en la plataforma, y a nivel de hardware la conexión es tan simple como conectar el pin VCC de alimentación a 3.3 o 5 (V), el pin GND a Tierra y el pin de datos a cualquier pin digital del microcontrolador. La lectura de los datos se realiza cada 2 segundos, y cada sensor viene calibrado de fábrica para proporcionar alta estabilidad y buena fiabilidad con el paso del tiempo [10].

Como recomendaciones del fabricante el hilo que comunica el sensor y el microcontrolador no debe ser mayor a 20 (m), se debe proteger al sensor de la luz directa del sol, y se debe utilizar una resistencia de 4.7 (K Ohm) en modo pull-up entre el pin de datos y el de alimentación [10]. En el caso del módulo DHT22 la resistencia no es necesaria, pues este sensor cuenta con una circuitería incorporada, donde se halla una resistencia en *pull-up* y un condensador entre el pin de datos y el de tierra, que actúa de filtro para interferencias en la parte de alimentación [11].

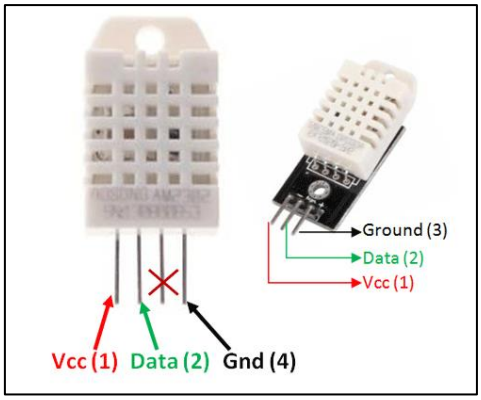


**Figura 1.3** Sensor y módulo DHT22 [11]

# 2 METODOLOGÍA

El desarrollo de este proyecto tiene como propósito el cumplimiento de cinco objetivos clave, que son esenciales para la implementación de un prototipo eficiente de transmisión y medición de datos. La metodología seleccionada para llevar a cabo este proyecto se ilustra en la Figura 2.1 y se ha desarrollado cuidadosamente para asegurar su éxito.

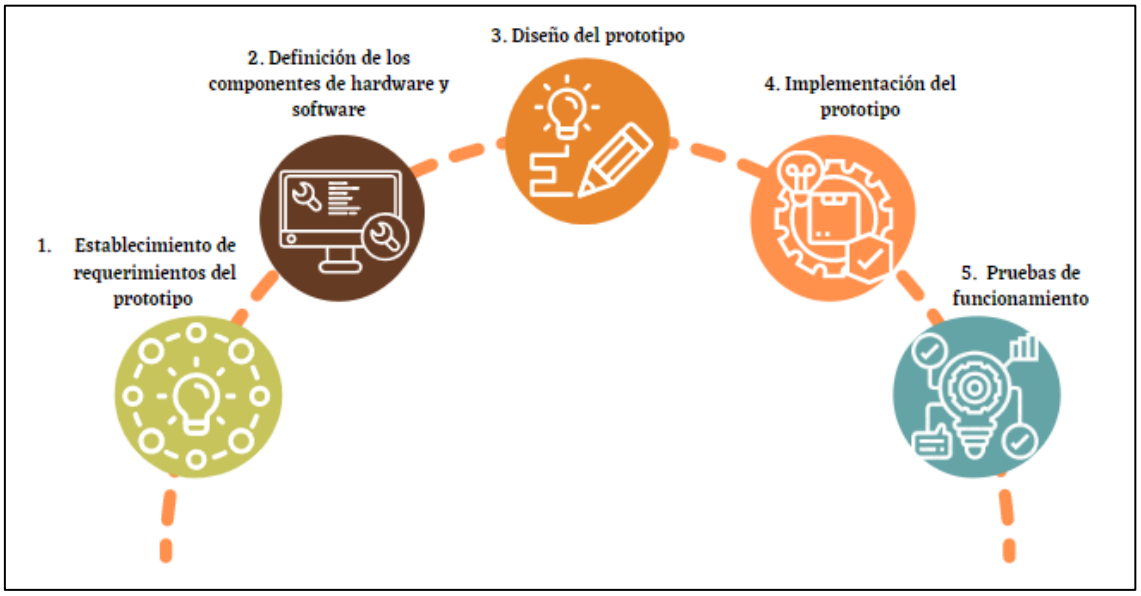


**Figura 2.1** Metodología utilizada

Se establecieron los requerimientos del equipo y lo que precisa para su correcto funcionamiento, entre ellos se encuentra la portabilidad, conectividad IoT con LoRa, lectura de datos exitosa, y protección del prototipo.

Para la definición de componentes del prototipo, se realizó un análisis tanto del hardware como del *software* necesario. Respecto a la selección de los componentes físicos se hizo una comparativa entre microcontroladores que tengan la capacidad de comunicarse por medio de la tecnología LoRa, y se aplicó el mismo proceso para la selección del sensor de humedad y temperatura más adecuado para el prototipo. En el caso de la selección del *software*, se utilizó Arduino IDE debido a la compatibilidad que presenta la placa para programar en esta aplicación.

En el diseño del prototipo se creó un circuito (modelo 2D) que conecta el módulo con el sensor y da paso a las lecturas correctamente. De igual forma, se elaboró un modelado 3D de cómo se vería el prototipo una vez se encuentre armado y soldado.

Se realizó un análisis del consumo energético que tendrían los dispositivos y cuál sería la mejor forma de alimentarlos. En esta etapa también se desarrolló el diagrama de flujo del código con el que el transmisor haría la lectura y transmisión de datos, y del código con el que el receptor los recibiría y mostraría en su pantalla.

La implementación del prototipo se inició creando el circuito de conexión de la placa en el programa *Fritzing*, que contaba con la imagen del módulo aplicado. Gracias al mismo programa, fue posible realizar el diseño de las pistas en base al circuito realizado previamente y con ello se pasó a la creación de la Placa de Circuito Impreso (PCB).

Por otro lado, se programó el código con el que el módulo transmisor realizaría las lecturas de los parámetros y enviaría dichos datos al módulo receptor; de la misma manera, se hizo el código del módulo receptor cuya función sería no solo recibir la información, sino imprimirla en la pantalla OLED.

A continuación, se elaboraron dos estuches para que protejan tanto a el emisor como al receptor, cada uno adaptado a su tamaño necesario. Luego, se hizo el montaje del prototipo en cada caja de protección.

Finalmente, se realizaron las respectivas pruebas de funcionamiento del prototipo, poniéndolo a prueba en diferentes casos (con y sin línea de vista) y distancias; a esto se agrega una comparativa de esos mismos datos recolectados, con información obtenida de un dispositivo externo capaz de medir los mismos parámetros.

# 3 RESULTADOS

En cuanto al funcionamiento, el prototipo para transmisión inalámbrica de temperatura y humedad basado en tecnología LoRa, permite realizar el envío de lecturas de dichos parámetros desde un nodo transmisor hacia un nodo receptor a grandes distancias. En el módulo emisor las lecturas de los datos son captadas con el sensor DHT22 y son procesadas en el microcontrolador Wi-Fi LoRa32, gracias al código previamente cargado; a continuación, los resultados se muestran en la pantalla OLED y a su vez se envían de manera periódica (cada 2 segundos) al nodo LoRa receptor, el cual imprime en su pantalla las lecturas de temperatura y humedad obtenidas, actualizándose constantemente dependiendo de los datos recibidos.

El prototipo cuenta con un estuche para cada dispositivo, con el fin de proteger a los módulos. También se provee una alimentación a través de baterías de litio para ambas partes. Gracias a esto, la medición de los parámetros se puede realizar en variedad de lugares y tipos de ambiente.

## 3.1 Requerimientos del prototipo

El modelo por implementar requiere tener comunicación inalámbrica IoT utilizando la tecnología LoRa, para ello se elige un microcontrolador con un chip LoRa que trabaje en la banda de 915 Mhz.

El prototipo debe ser capaz de leer la temperatura y humedad relativa de cualquier ambiente al que sea expuesto, y debe presentar una visualización de estos datos. Por otra parte, los dispositivos necesitan ser portátiles para poder realizar mediciones en cualquier lugar; para ello, es preciso que la alimentación se provea por medio de una batería y que ambos módulos sean protegidos por estuches, con el fin de evitar daños a los nodos.

En cuanto al emisor, debido a que existe una conexión entre el sensor de temperatura y humedad y el microcontrolador, es necesaria la implementación de una placa electrónica que permita la conexión entre ambos dispositivos. En el caso del receptor no es necesario debido a que este solo necesita conexión a una batería.

Finalmente, para probar la validez de las lecturas realizadas con el prototipo, se debe comparar los datos obtenidos y transmitidos con la información proporcionada por un aparato externo capaz de medir la temperatura y la humedad ambiental; en este caso se trabaja con un termohigrómetro digital.

## 3.2 Definición de los componentes de *hardware* y *software*

### Hardware

La elección del microcontrolador con el que se va a trabajar en el prototipo se basa principalmente en las características que cada dispositivo provee y si estas son adecuadas para la implementación. Debido a que la comunicación debe realizarse utilizando tecnología LoRa, se plantea el uso de un transceptor SX127x conectado a un Arduino UNO o una ESP32 y una pantalla OLED; pero, ya que existen placas que cuentan con todas estas particularidades juntas, las opciones se reducen a los módulos mostrados en la Tabla 3.1 para mayor facilidad de implementación.

**Tabla 3.1** Comparativa de módulos LoRa [8] [12]

| Parámetro | Wi-Fi LoRa 32 (V2) | TTGO LoRa32 |
|---|---|---|
| Marca | Heltec | LILYGO |
| Chip maestro | ESP32 Tensilica LX6 | ESP32 PICO-D4 |
| Voltaje de operación | 3.3 ~ 4.2 (V) | 1.8 ~ 3.7 (V) |
| Frecuencia de trabajo | 863 ~ 923 (MHz) | 868 ~ 915 (MHz) |
| Temperatura de operación | -20 ~ 70 (°C) | -40 ~ +85 (°C) |
| Potencia de transmisión | 19 (dBm) ± 1 (dBm) | +20 (dBm) |
| Sensibilidad de recepción | -135 (dBm) | -139 / -136 / -118 / -123 (dBm) |
| Dimensiones | 51 x 25.5 x 10.6 (mm) | 64.47 x 27 x 8.54 (mm) |
| Pantalla OLED | 2.44 (cm)<br>128×64 píxeles | 2.44 (cm)<br>128×64 píxeles |
| Conector de batería | SH1.25 x2 socket | JST XH2-2.54mm |
| Memoria | 8 (MB) | 4 (MB) |
| Compatibilidad con Arduino IDE | Sí | Sí |
| Diseño | | |

A pesar de que ambos módulos presentan características casi iguales, el dispositivo seleccionado es el Wi-Fi Lora 32. Dicho microcontrolador cuenta con una memoria más grande que el TTGO LoRa 32, y tiene un tamaño más compacto. Sus voltajes de operación resultan más acordes con el manejo del sensor de temperatura y humedad empleado y presenta una sensibilidad de recepción coherente para trabajar como un nodo. Asimismo, la frecuencia de trabajo que maneja tiene un rango más amplio (863 ~ 923 [MHz]) que el rango del TTGO LoRa32 (868 ~ 915 [MHz]) [8]; lo que resulta ventajoso pues la frecuencia central en la que se trabaja es de 915 (MHz), por lo que en el caso del TTGO se encontraría en el borde del rango y podría presentar complicaciones en la comunicación [12].

En cuanto a la elección del sensor necesario para la lectura de los parámetros requeridos, se tienen dos opciones: DHT11 y DHT22. Ambos siendo de la misma gama, pero con características distintas que se presentan en la Tabla 3.2.

**Tabla 3.2** Comparativa de sensores de humedad y temperatura [13] [14]

| **Parámetros** | **DHT11** | **DHT22** |
|---|---|---|
| Voltaje de Operación | 3 – 5 (V) DC | 3 – 6 (V) DC |
| Rango de medición de temperatura | 0 a 50 (°C) | -40°C a 80 (°C) |
| Precisión de medición de temperatura | ±2.0 (°C) | <±0.5 (°C) |
| Resolución Temperatura | 0.1 (°C) | 0.1 (°C) |
| Rango de medición de humedad | 20 a 90 (%RH) | De 0 a 100 (%RH) |
| Precisión de medición de humedad | 4 (%RH) | 2 (%RH) |
| Resolución Humedad | 1 (%RH) | 0.1 (%RH) |
| Tiempo de sensado | 2 seg | 2 seg |

El sensor seleccionado es el DHT22, por poseer un rango de medición más alto y una mejor precisión al medir la temperatura y la humedad. Por otra parte, el modelo DHT22 cuenta con dos tipos: sensor y módulo. Como se aprecia en la Figura 3.1, el módulo LoRa cuenta con un pequeño circuito incorporado, cuya ventaja es que no se necesita trabajar con ninguna resistencia externa pues ya la tiene implementada de antemano; por mayor comodidad se elige el módulo DHT22 [15].

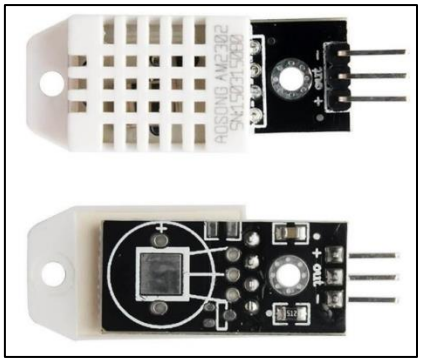

**Figura 3.1** Módulo DHT22 [15]

### Software

El *software* seleccionado para realizar la programación de los códigos de transmisión y recepción es Arduino IDE. El Entorno de desarrollo integrado (IDE), es un *software* gratuito y de código abierto que permite a los usuarios programar y cargar código en la placa Arduino, u otras placas similares. Es una herramienta útil para los aficionados y los profesionales de la electrónica, ya que facilita el proceso de desarrollo de proyectos que utilizan microcontroladores [16].

El *software* es compatible con una variedad de plataformas y sistemas operativos, incluyendo Windows, Mac OS y Linux. Proporciona un entorno de programación fácil de usar, que incluye una interfaz gráfica de usuario (GUI) y una serie de funciones y herramientas para simplificar el proceso de desarrollo. Además, incluye una gran cantidad de librerías y ejemplos de código, lo que facilita la tarea de escribir programas para controlar sensores, actuadores y otros componentes electrónicos. [16].

Se selecciona este programa porque los módulos con los que trabaja el prototipo tienen compatibilidad con IDE; además de que es una herramienta práctica, intuitiva y sencilla de manejar [16].

## 3.3 Diseño del prototipo

### Circuito electrónico

Su diseño se realizó utilizando *Fritzing*. Este programa, además de proporcionar una vista en placa de pruebas, un esquemático y el diseño de PCB, cuenta con la imagen del módulo Wi-Fi LoRa 32, lo que permite realizar las conexiones de una manera sencilla sin recurrir al uso de otras placas para aparentar sus pines.

En la Figura 3.2 se puede observar cómo se vería la conexión entre el microcontrolador y el sensor. Se utiliza un sensor de 4 pines únicamente como reemplazo del módulo

DHT22 en la aplicación. En la Figura 3.3 se aprecia la forma del circuito y sus conexiones respectivas para cada pin.

Cabe recalcar que, como se mencionó previamente en la selección de *hardware*, no se hizo uso de ninguna resistencia debido a que el sensor de humedad y temperatura ya cuenta con una implementada internamente.

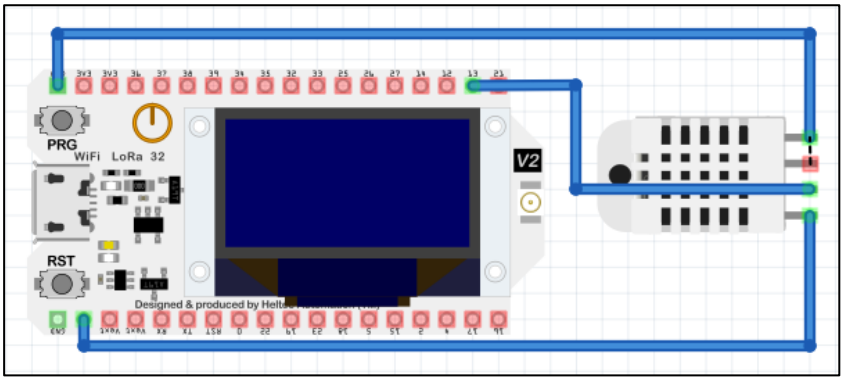


**Figura 3.2** Conexión entre el módulo y el sensor

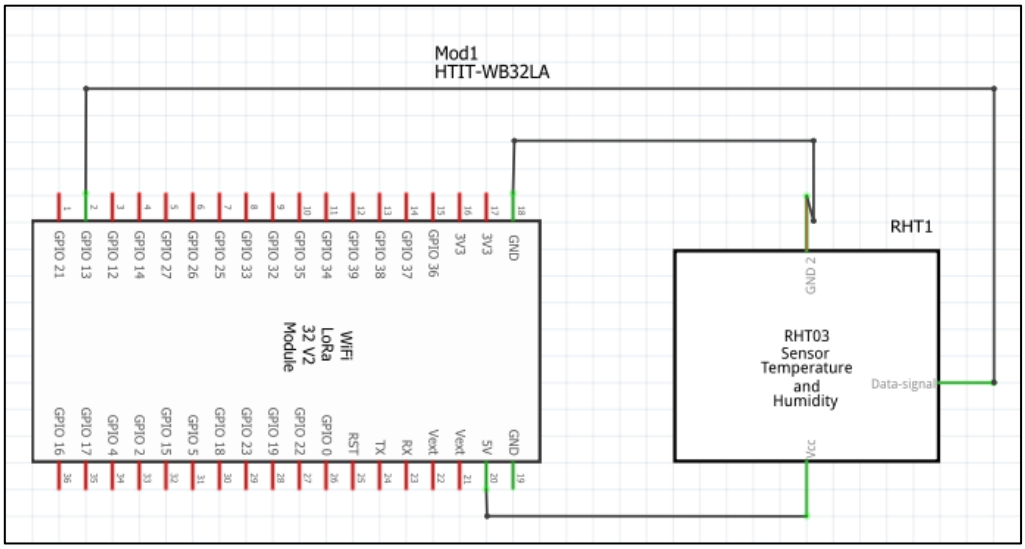


**Figura 3.3** Circuito electrónico

### Consumo energético

Se presenta a continuación la Tabla 3.3, donde se detalla el consumo de cada equipo utilizado:

**Tabla 3.3** Consumo energético [8] [10] [17]

| Parámetros | LoRa 32 | Sensor DHT22 |
|---|---|---|
| Voltaje de alimentación | 5 (V) USB / 3.7 (V) Batería | 3.3 – 5.5 (V) |
| Corriente de operación | ≥500 (mA) [USB] / ≥250 (mA) [Batería] | 2.5 (mA) |

La alimentación del equipo se elige tomando en cuenta que se desea que el prototipo sea portable, por lo tanto, el uso de una batería es la opción ideal. Por recomendación del fabricante, se utiliza una batería de litio de 3.7(V) conectada a la interfaz de batería integrada SH1.25-2 [8].

La alimentación del sensor se realiza conectando el pin Vcc del sensor al pin de 5(V) del microcontrolador, tomando en cuenta que el voltaje de alimentación más común para el sensor es de 5(V).

### Diseño 3D del prototipo

Se realiza un diseño 3D de cómo se vería el prototipo finalizado. El programa en el que se modelaron las placas es *SketchUp*. Debido a una falta de la imagen del Heltec WiFi LoRa32 se efectúa un reemplazo de la misma por una ESP32 disponible en la aplicación, exclusivamente para visualizar la forma en la que se vería el microcontrolador.

Para el módulo transmisor, se coloca el sensor de humedad y temperatura correspondiente, el microcontrolador, la batería de litio y se grafica una caja de protección, como se presenta en la Figura 3.4. En cuanto al dispositivo receptor, se coloca únicamente el microcontrolador y una batería de litio que lo alimenta, todo dentro de su respectiva caja de protección, como se observa en la Figura 3.5.

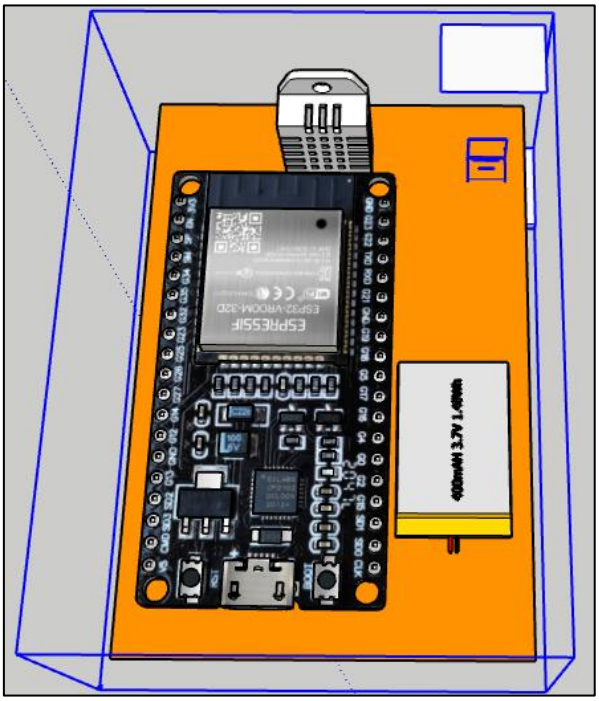


**Figura 3.4** Modelo 3D del módulo transmisor

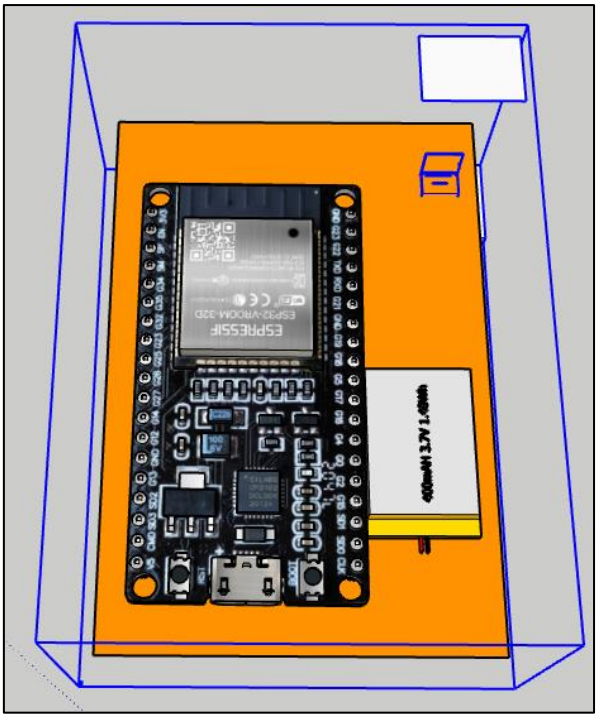


**Figura 3.5** Modelo 3D del módulo receptor

### Diagrama de flujo

<u>Módulo transmisor</u>

El código fuente del módulo transmisor contiene dos partes vitales: la lectura de datos mediante el sensor y envío de datos desde el módulo, como se presenta en la Figura 3.6. Se inicia con una inclusión de todas las librerías que se necesitan para el funcionamiento de la pantalla OLED, el módulo y el sensor; seguido por una definición de los pines y la banda de frecuencia en los que van a trabajar. Se crea una función "dht" donde se especifica el pin de trabajo y el tipo de sensor utilizado.

Como siguiente paso se crea la función principal (que se ejecuta una sola vez), donde se inicializa la pantalla, el módulo LoRa y el sensor; mostrando el mensaje pertinente que indique que el dispositivo se encuentra listo para utilizar.

A continuación, se pasa a la función *loop*, es decir aquella que se repite cada cierto tiempo. Aquí se comienza con la obtención de datos del sensor DHT22, si estas lecturas son correctas se imprimen en la pantalla OLED, caso contrario se imprime un mensaje que indique que existe un fallo en la lectura. Adicional a esto, los datos recolectados se envían al LoRa receptor y el proceso se repite.

Cabe recalcar que, de darse algún error en la lectura, este no tendrá que ver con el código; sino que debe ser solucionado físicamente, ya que se debe a algún problema en el contacto entre el sensor y el módulo.

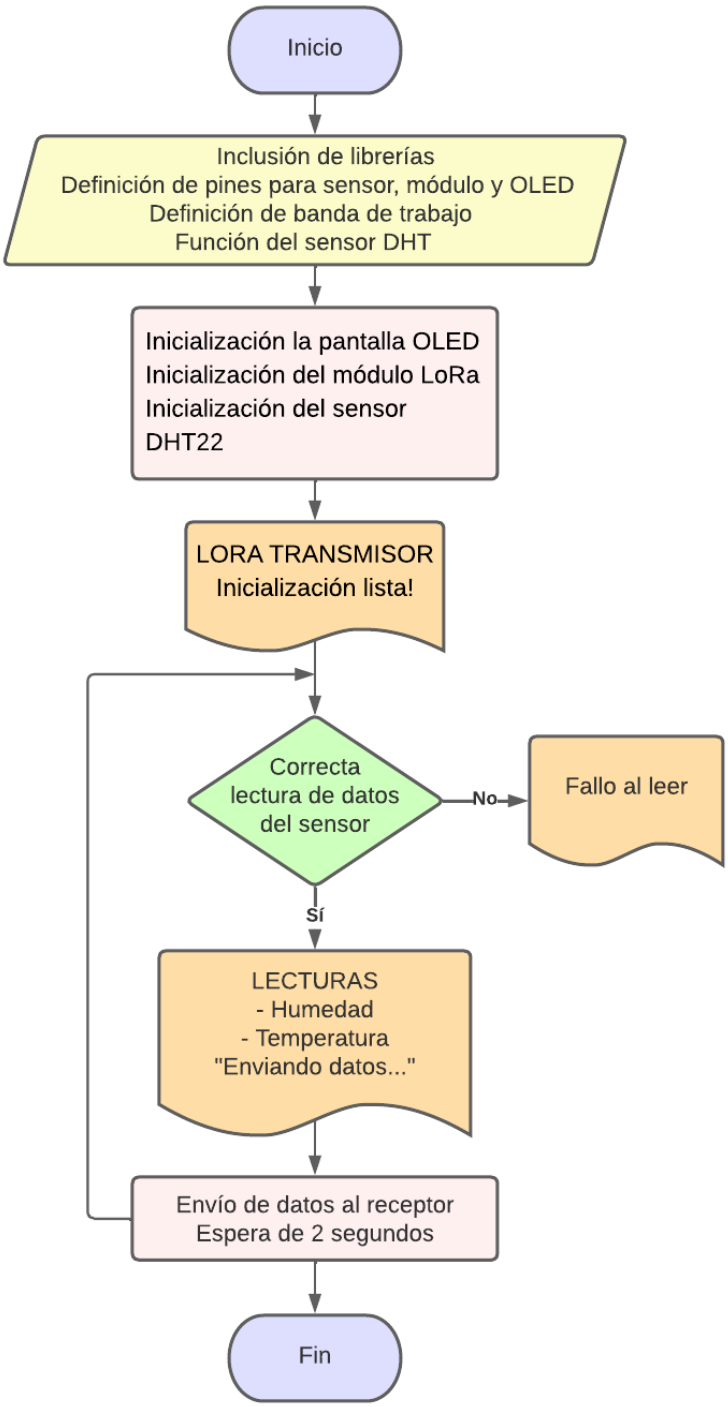


**Figura 3.6** Diagrama de flujo del módulo transmisor

Módulo receptor

Como se observa en la Figura 3.7, en el receptor el proceso es similar al del transmisor. Se inicia con una inclusión de las librerías necesarias, una definición de los pines y la banda con los que se va a trabajar; pero, adicionalmente se incluye la librería que maneja la fuente con la que se va a imprimir en la pantalla OLED, esto con el fin de que las lecturas se puedan visualizar claramente.

En la función principal se inicializa la pantalla y el módulo LoRa; mostrando el mensaje pertinente que indique que el dispositivo se encuentra listo para utilizar. Además, se incluye la función *string*, misma que permite la recepción de los datos. Una vez listo este proceso se imprime un mensaje en la pantalla que indica que el dispositivo se encuentra listo para utilizarse. Como siguiente paso se tiene la función *loop*, donde se ejecuta la recepción e impresión de las lecturas en la pantalla, repitiendo el proceso cada vez que reciba información nueva.

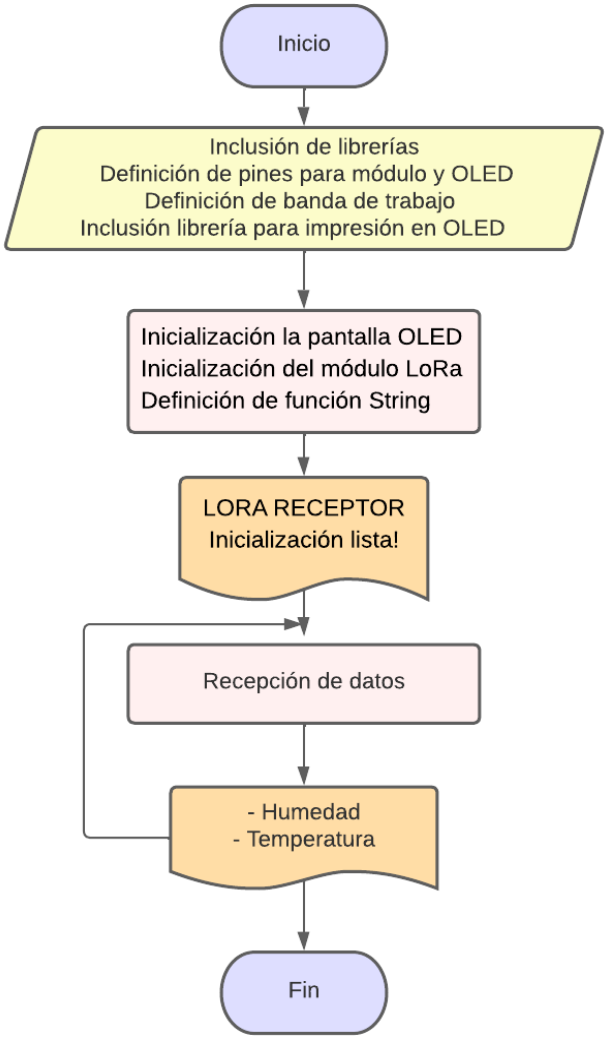


**Figura 3.7** Diagrama de flujo del módulo receptor

## 3.4 Implementación del prototipo

### Elaboración de la placa electrónica

Para elaborar la PCB se inicia dibujando las pistas del circuito en un *software*; en este caso, como se mencionó anteriormente, se utiliza la herramienta *Fritzing*, el resultado se muestra en la Figura 3.8. La vista que se presenta en la Figura 3.9 se obtiene exportando el circuito en PDF para imprimirlo y es la que se utiliza para traspasar las pistas a la baquelita.

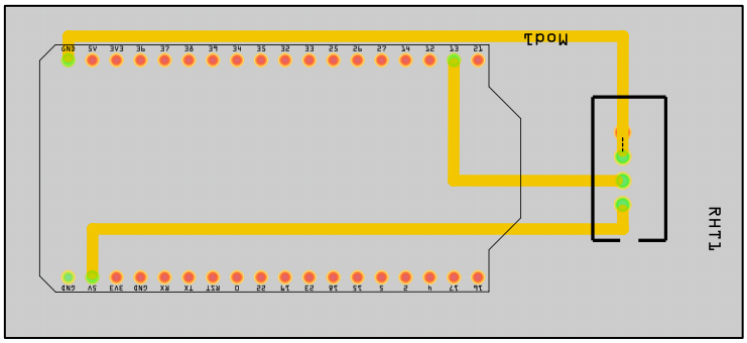


**Figura 3.8** Pistas del circuito en el programa

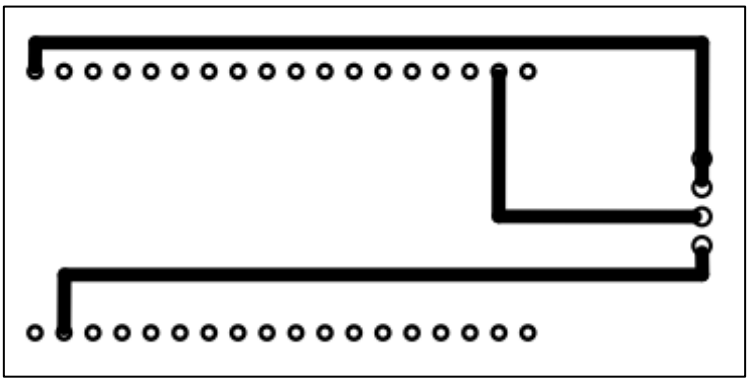

**Figura 3.9** Pistas del circuito

Se empieza puliendo la baquelita con una esponjilla de lana de acero, para quitar cualquier tipo de grasa o suciedad y dejarla brillante. El método utilizado para dibujar las pistas es el “método del marcador”, que consiste en dibujar manualmente las líneas con un marcador permanente negro. Esto se realiza en una baquelita de 7 x 5 (cm) como se observa en la Figura 3.10.

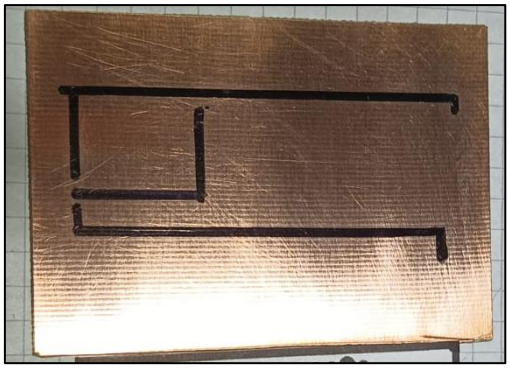

**Figura 3.10** Circuito dibujado en la baquelita pulida

A continuación, como se logra ver en la Figura 3.11 se coloca ácido férrico en una tarrina y se sumerge hasta que se consuma completamente el cobre, lo único que debe quedar son las pistas dibujadas.

**Figura 3.11** Baquelita sumergida en cloruro férrico

Una vez se deshace todo el cobre, se saca la baquelita y se la seca con una toalla; se pule el marcador negro con la esponjilla y el resultado es el que se muestra en la Figura 3.12.

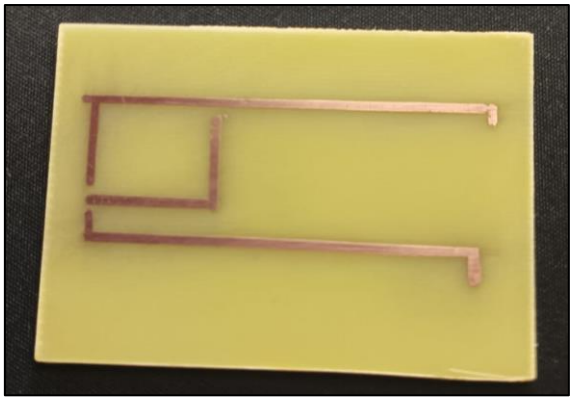

**Figura 3.12** Pista de cobre

Ya con la pista lista, se realizan los agujeros correspondientes para encajar las piezas en la baquelita; para ello, se utiliza una broca de 1.0 (mm) y un taladro manual. Posteriormente, se colocan y sueldan todas las piezas a la baquelita. Este proceso se aprecia en la Figura 3.13.

**Figura 3.13** Perforación y soldadura de la placa

Adicionalmente, se agrega un interruptor entre el módulo y la batería para encender y apagar el dispositivo. El resultado final de la placa se muestra en la Figura 3.14.

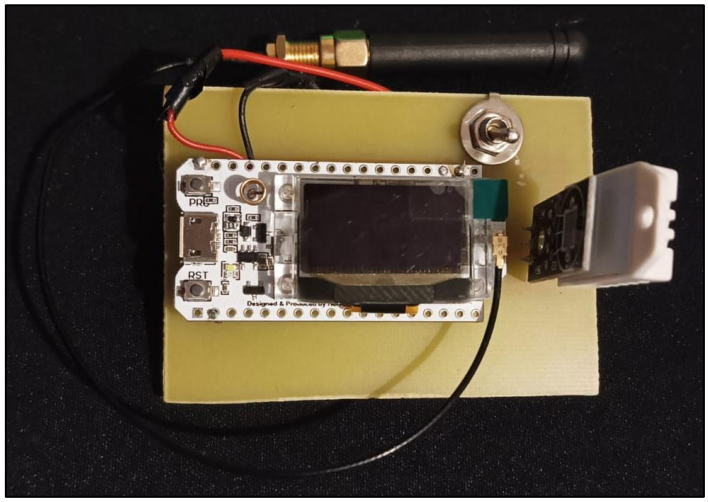


**Figura 3.14** PCB final

### Elaboración de las cajas de protección

Se elaboran cajas para el prototipo debido a que sus partes serán expuestas a diferentes ambientes y por recomendación del fabricante el sensor debe encontrarse cubierto y protegido de la luz directa del sol (radiación UV) [10].

El diseño de la caja del dispositivo transmisor se realiza en el programa *AutoCAD*, ahí se dibujan los planos y las medidas de cada parte, el plano final se puede observar en la Figura 3.15. Este plano se exporta en formato xdf para posteriormente realizar los cortes de cada pieza; el material elegido es el acrílico y el resultado final de la caja de protección se visualiza en la Figura 3.16.

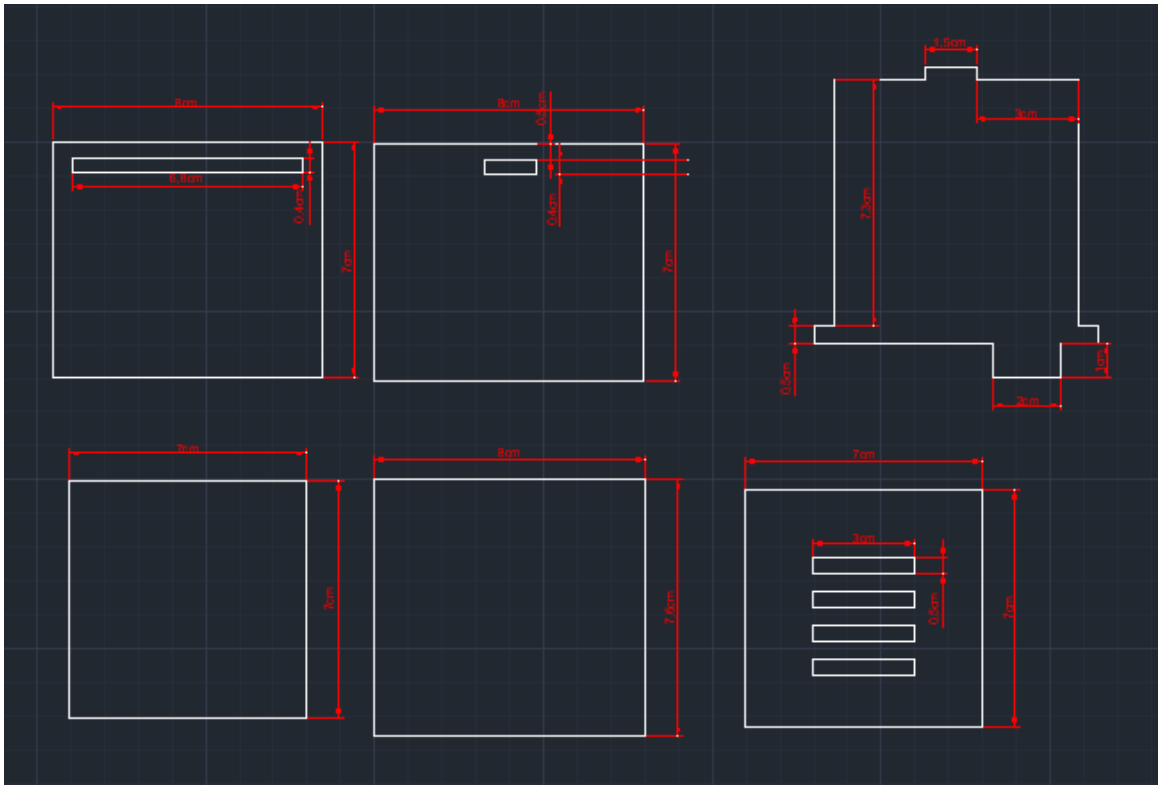


**Figura 3.15** Plano caja de protección en AutoCAD

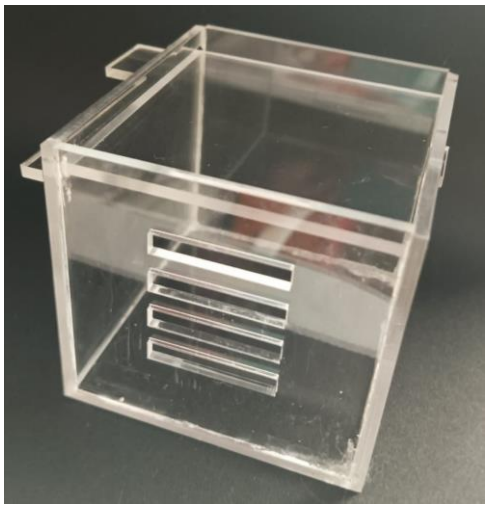

**Figura 3.16** Caja de protección resultante

Para el estuche de protección del dispositivo receptor se utiliza una caja transparente pequeña y se agrega un agujero para encajar la antena, como se observa en la Figura 3.17. Adicionalmente, se personaliza la caja para que el cableado interno no sea visible, el resultado se presenta en la Figura 3.18.

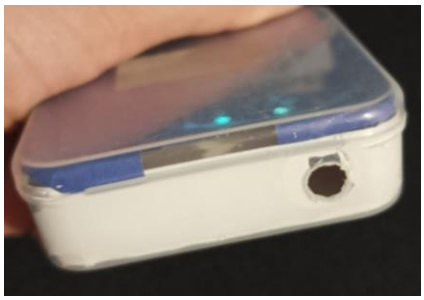

**Figura 3.17** Agujero para la antena

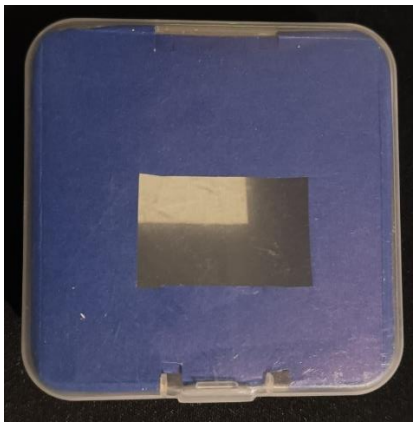

**Figura 3.18** Caja de protección del módulo receptor

### Programación de los módulos WiFi LoRa 32

La programación, desarrollo y compilación de los códigos se realizan en Arduino IDE. Se tienen dos códigos: módulo transmisor y módulo receptor. Nótese la similitud en ambos códigos, tomando en cuenta que el uso de librerías, definición de pines y manejo de la pantalla son iguales.

<u>Código del módulo transmisor</u>

Como se presenta en la Figura 3.19, se inicia incluyendo todas las librerías necesarias para el funcionamiento de cada parte del circuito (LoRa, pantalla OLED y sensor); de la misma manera, se definen los pines de trabajo del sensor, del módulo transceptor y de la pantalla, así como el tipo de sensor que se va a utilizar y la frecuencia de trabajo del dispositivo, tomando en cuenta que por la región en la que se encuentra el módulo la frecuencia de trabajo es de 915 (MHz). Adicionalmente se define el ancho y alto de la pantalla OLED integrada.

```
//Inclusión de librerías para LoRa
#include <SPI.h>
#include <LoRa.h>

//Inclusión de librerías para la pantalla OLED
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//Inclusión de librerías para el sensor
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 13       //Definición del pin por el que van a ingresar los datos
#define DHTTYPE DHT22   //Definición del tipo de sensor DHT con el que se va a trabajar

//Definición de los pines utilizados por el módulo transceptor LoRa
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26

//Definición de la frecuencia a la que va a trabajar el módulo
#define BAND 915E6

//Definición de los pines con los que va a trabajar la pantalla OLED
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16

//Definición del tamaño de la pantalla OLED
#define SCREEN_WIDTH 128 //Ancho de la pantalla OLED, en pixeles
#define SCREEN_HEIGHT 64 //Alto de la pantalla OLED, en pixeles
```

**Figura 3.19** Inclusión de librerías y definición de pines

A continuación, se crea un objeto de *Adafruit* con el nombre de *display*; este objeto maneja todo lo relacionado al uso de la pantalla OLED. También se define una función *dht* para el sensor, donde se llama al pin de datos y al sensor DHT especificado previamente, como se indica en la Figura 3.20.

```
//Creación de un objeto Adafruit_SSD1306 llamado "display", con el que se trabaja todo lo que tiene que ver con el OLED
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

//Definición de una función "dht" donde se especifica el pin de trabajo y el tipo de sensor
DHT dht(DHTPIN, DHTTYPE);
```

**Figura 3.20** Objeto display y función dht

Posteriormente, se da paso a la función principal *setup*, donde se realiza todo el proceso de inicialización y preparación del LoRa32. Como se aprecia en la Figura 3.21, se comienza con un reinicio vía *software* de la pantalla OLED y luego se pasa a su inicialización. Se limpia la pantalla, se elige el color, tamaño y coordenadas del texto, y se imprime el nombre del módulo: “LORA TRANSMISOR”.

```
//Función principal, donde se inicializa todo
void setup() {

  //Reseteo de la pantalla OLED via software para tener un inicio correcto
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  //Inicialización la pantalla OLED
  Wire.begin(OLED_SDA, OLED_SCL);   //Se inicia una comunicación I2C usando los pines OLED_SDA y OLED_SCL y la función Wire.begin()
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) {
    display.clearDisplay();
    display.setCursor(0,0);
    display.println(F("SSD1306 allocation failed"));  //Mensaje en caso de que el OLED no se inicialice correctamente
    for(;;); //Con el for se indica que no proceda y cree un bucle permanente en caso de fallo
  }

  display.clearDisplay();           //Se limpia la pantalla
  display.setTextColor(WHITE);      //Se elige el color del texto
  display.setTextSize(1);           //Se elige el tamaño del texto
  display.setCursor(0,0);           //Se elige las coordenadas de la pantalla donde se imprimirá el texto
  display.print("LORA TRANSMISOR"); //Título inicial
  display.display();                //Se muestra en la pantalla OLED
```

**Figura 3.21** Inicialización de pantalla y formato de texto

En el siguiente paso se definen los pines *Serial Peripheral Interface* (SPI) que utiliza el chip LoRa y se configura al módulo transceptor. Se continúa con la inicialización del módulo, y en caso de existir un fallo al prenderlo se muestra el mensaje “Inicio de LoRa fallido!”; por otra parte, en caso de que este sea correcto el mensaje que se imprime es el de “Inicialización lista!”, tal y como se muestra en la Figura 3.22. Se finaliza la función principal con la función de inicialización del sensor DHT.

```
  //Se define los pines SPI usados por el chip LoRa
  SPI.begin(SCK, MISO, MOSI, SS);
  //Se configura el módulo transceptor LoRa
  LoRa.setPins(SS, RST, DIO0);

  //Inicialización del módulo LoRa utilizando la función begin() en el objeto "LoRa" y "BAND" como argumento
  if (!LoRa.begin(BAND)) {
    display.clearDisplay();
    display.setCursor(0,0);
    display.println("Inicio de LoRa fallido!"); //Mensaje en caso de tener un inicio fallido del módulo
    display.display();
    while (1);
  }

  display.setCursor(0,10);
  display.print("Inicializacion lista!"); //Mensaje de inicialización de LoRa
  display.display();
  dht.begin();        //Inicialización del sensor DHT22
}
```

**Figura 3.22** Inicialización del módulo transceptor

La función *loop* presentada en la Figura 3.23 es la función que se va a repetir de manera constante para la obtención y envío de datos. El primer paso es añadir un lapso de dos segundos de espera antes de que se repita.

Luego, se obtienen las lecturas de humedad y temperatura del sensor gracias a las funciones *dht.readHumidity* y *dht.readTemperature*; estos datos se guardan como números flotantes (decimales).

En caso de que exista un fallo en la lectura de datos, se escribe en la pantalla el mensaje “Fallo al leer”.

```
//Función loop, donde se realiza la lectura y envío de datos
void loop() {

  delay(2000);    //Se señala un tiempo de espera de 2 segundos antes de la siguiente lectura y envío

  float h = dht.readHumidity();     //Obtención de las lecturas de humedad del sensor
  float t = dht.readTemperature();  //Obtención de las lecturas de temperatura del sensor

  //En caso de que no haya una correcta lectura de datos se muestra un mensaje que indique el fallo
  if(isnan(h)||isnan(t)) {
  display.clearDisplay();
  display.setCursor(0,0);
  display.println("Fallo al leer");  //Mensaje de fallo detectado
  display.display();
  return;
 }
```

**Figura 3.23** Función *loop*, obtención de lecturas

La parte final de esta función se observa en la Figura 3.24, donde se realiza la impresión de las lecturas en el OLED y el envío de los paquetes LoRa al receptor en forma de cadenas *string*. Cabe aclarar que el texto que aparece en la pantalla del receptor se escribe en este código.

```
//Impresión en la pantalla OLED de las lecturas
  display.clearDisplay();
  display.setCursor(0,0);
  display.println("LECTURAS");  //Título

  display.setCursor(0,20);      //Impresión de datos de Humedad
  display.print("Humedad:");
  display.setCursor(50,20);
  display.print(h);

  display.setCursor(0,30);      //Impresión de datos de Temperatura
  display.print("Temperatura:");
  display.setCursor(80,30);
  display.print(t);

  display.setCursor(0,50);
  display.print("Enviando datos...");  //Mensaje de envío de datos
  display.display();

//Envío de los paquetes LoRa
  LoRa.beginPacket();   //Inicialización del paquete LoRa
  LoRa.print("H:"+String(dht.readHumidity(),1)+"%\n");      //Envío de datos de Humedad
  LoRa.print("T:"+String(dht.readTemperature(),1)+"*C\n");  //Envío de datos de Temperatura
  LoRa.endPacket();     //Finalización del paquete LoRa
}
```

**Figura 3.24** Impresión de lecturas y envío de las mismas hacia el receptor

Código del módulo receptor

El código del módulo receptor es muy similar al del módulo transmisor, por lo que a continuación se explica únicamente los cambios o adiciones realizados al mismo, siendo el cambio principal la exclusión completa de todas las funciones y librerías relacionadas con el sensor DHT.

Como se observa en la Figura 3.25, en la inclusión de librerías se agrega una nueva para el uso de una fuente de texto distinta a la que viene por defecto.

```
//Inclusión de fuente para el texto que se imprime en el OLED
#include <Fonts/FreeMono12pt7b.h>
```

**Figura 3.25** Librería para fuente de texto

Se define un objeto *string*, cuyo cargo es el de representar y manipular una secuencia de caracteres, en este caso, las lecturas recibidas, que se aprecian en la Figura 3.26.

```
//Definición de la cadena string llamada "LoRaData", donde se guarda la información recibida
String LoRaData;
```

**Figura 3.26** Objeto string

En la función *setup* el único cambio significativo se encuentra en el nombre del módulo, que en esta ocasión es “LORA RECEPTOR” (Figura 3.27). Todas las demás inicializaciones se realizan normalmente, exceptuando la del sensor.

```
display.clearDisplay();          //Se limpia la pantalla
display.setTextColor(WHITE);     //Se elige el color del texto
display.setTextSize(1);          //Se elige el tamaño del texto
display.setCursor(0,0);          //Se elige las coordenadas de la pantalla donde se imprimirá el texto
display.print("LORA RECEPTOR");  //Título inicial
display.display();               //Se muestra en la pantalla OLED
```

**Figura 3.27** Cambio en función *setup*

En cuanto a la función *loop,* esta sí presenta un cambio completo. Con la ayuda de la función “LoRa.parsePacket()”, se averigua si hay paquetes nuevos y con “LoRa.readString” se lee su contenido. Finalmente, se imprime en la pantalla las lecturas, especificando la posición y fuente con la que se desea mostrarlas, como se aprecia en la Figura 3.28.

```
void loop() {

  int packetSize = LoRa.parsePacket();  //Función parsePacket() para averiguar si hay algún paquete nuevo
  if (packetSize) {                     //Recepción del paquete
    while (LoRa.available()) {
      LoRaData = LoRa.readString();     //Lectura del paquete con ayuda del método readString()
    }

    //Impresión de los datos en la pantalla OLED
    display.clearDisplay();
    display.setCursor(0,20);
    display.setFont(&FreeMono12pt7b); //Se define la fuente con la que se va a imprimir el texto en la pantalla OLED
    display.print(LoRaData);          //Impresión de los datos recibidos
    display.display();
  }
}
```

**Figura 3.28** Recepción e impresión de la información

### Montaje del prototipo en las cajas de protección

Para finalizar el transmisor, se encaja la placa cuidando que el sensor se encuentre del lado de la rejilla. Se colocan pilares en la parte de abajo para evitar cualquier movimiento indeseado y se cierra. El producto final es el que se muestra en la Figura 3.29 y en la Figura 3.30.

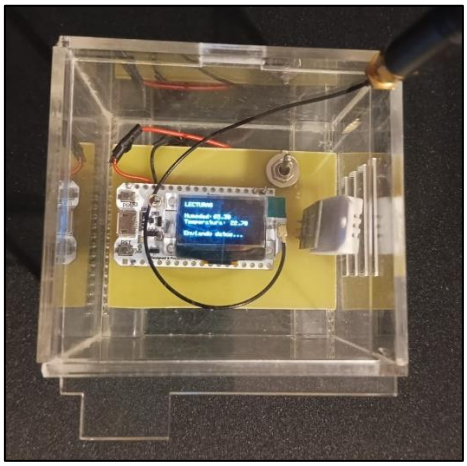

**Figura 3.29** Vista superior

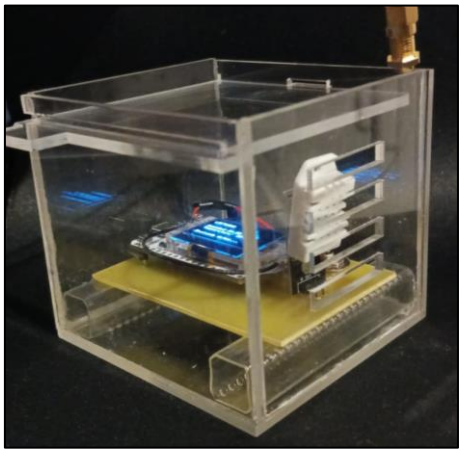

**Figura 3.30** Vista de caras laterales

En cuanto al receptor, se monta el módulo LoRa junto con la batería y su respectiva antena, y el resultado final es el que se muestra en la Figura 3.31.

**Figura 3.31** Módulo receptor en la caja de protección

## 3.5 Pruebas de funcionamiento

El cálculo de error entre las mediciones se realiza con la siguiente fórmula [18]:

$$\%Error = \frac{|E - T|}{T} \times 100$$

Donde:

E representa el valor experimental,

T representa el valor teórico.

En el caso de los errores en la medición de temperatura y humedad, el valor experimental representa a las lecturas del prototipo, mientras que el valor teórico representa a las medidas obtenidas con un termohigrómetro digital.

En el caso de los errores en los resultados de transmisión, el valor experimental representa a las lecturas recibidas en el receptor, mientras que el valor teórico representa a las lecturas enviadas por el transmisor.

### Resultados medición de temperatura y humedad

Para las pruebas de funcionamiento de las medidas obtenidas con el sensor, se plantean cuatro escenarios distribuidos en varias horas del día y con dos termohigrómetros diferentes. Esto con el propósito de validar las lecturas que proporciona el prototipo.

**Escenario1:** Jardín externo con clima soleado. Los resultados de la comparativa se aprecian en la Tabla 3.4; mientras que la evidencia de las lecturas del prototipo se muestra en la Figura 3.32, y del termohigrómetro digital TH-3 en la Figura 3.33.

**Tabla 3.4** Resultados escenario 1

| Escenario 1 // 11:00 a.m. | | | |
|---|---|---|---|
| **Parámetro** | **Prototipo** | **Termohigrómetro** | **Error (%)** |
| **Temperatura (°C)** | 29.0 | 28.2 | 2.84 |
| **Humedad (%)** | 44.9 | 47.5 | 5.47 |

**Figura 3.32** Escenario 1 - prototipo

**Figura 3.33** Escenario 1 – termohigrómetro digital

**Escenario 2:** Instalaciones de la CICAM, dentro del laboratorio. Los resultados de la comparativa se aprecian en la Tabla 3.5; mientras que la evidencia de las lecturas del prototipo se muestra en la Figura 3.34, y del termohigrómetro digital Taylor 1732 en la Figura 3.35.

**Tabla 3.5** Resultados escenario 2

| Escenario 2 // 14:20 p.m. | | | |
|---|---|---|---|
| **Parámetro** | **Prototipo** | **Termohigrómetro** | **Error (%)** |
| **Temperatura (°C)** | 22.2 | 21.5 | 3.26 |
| **Humedad (%)** | 51.2 | 49.0 | 4.49 |

**Figura 3.34** Escenario 2 - prototipo

**Figura 3.35** Escenario 2 – termohigrómetro digital

**Escenario 3:** Instalaciones de la CICAM, fuera del edificio. Los resultados de la comparativa se aprecian en la Tabla 3.6; mientras que la evidencia de las lecturas del prototipo se muestra en la Figura 3.36, y del termohigrómetro digital Taylor 1732 en la Figura 3.37.

**Tabla 3.6** Resultados escenario 3

| **Escenario 3 // 14:35 p.m.** | | | |
|---|---|---|---|
| **Parámetro** | **Prototipo** | **Termohigrómetro** | **Error (%)** |
| **Temperatura (°C)** | 22.7 | 22.0 | 3.18 |
| **Humedad (%)** | 44.6 | 41.0 | 8.78 |

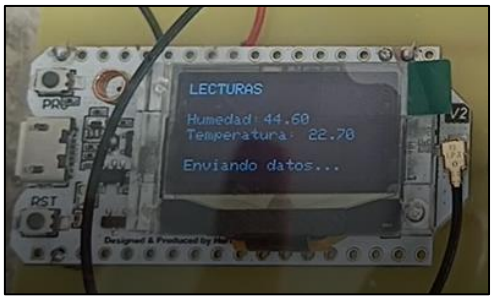


**Figura 3.36** Escenario 3 - prototipo

**Figura 3.37** Escenario 3 – termohigrómetro digital

**Escenario 4**: Jardín externo con clima templado, en la noche. Los resultados de la comparativa se aprecian en la Tabla 3.7; mientras que la evidencia de las lecturas del prototipo se muestra en la Figura 3.38, y del termohigrómetro digital TH-3 en la Figura 3.39.

**Tabla 3.7** Resultados escenario 4

| **Escenario 4 // 21:00 p.m.** | | | |
|---|---|---|---|
| **Parámetro** | **Prototipo** | **Termohigrómetro** | **Error (%)** |
| **Temperatura (°C)** | 18.3 | 17.9 | 2.23 |
| **Humedad (%)** | 75.7 | 74.1 | 2.16 |

**Figura 3.38** Escenario 4 – prototipo

**Figura 3.39** Escenario 4 – termohigrómetro digital

Como se puede observar, el error máximo de temperatura se encuentra en la Tabla 3.5 con un 3.26% de error; mientras tanto, el error máximo de humedad se encuentra en la Tabla 3.6, con un error del 8.78%. Ambos porcentajes de error se encuentran por debajo del 10%, por lo que se cumple con el error máximo permitido para saber si las lecturas del sensor son correctas; con esto se comprueba la validez de la información proporcionada por el prototipo.

Adicionalmente, al hacer una comparativa entre los errores de humedad y de temperatura, se observa que los errores de humedad son más variables mientras que los de temperatura se mantienen más estables, esto puede ser causado por los tipos de termohigrómetros con los que se hizo la comparativa y la caja de protección del prototipo.

### Resultados de transmisión de datos

Para las pruebas de funcionamiento de transmisión de datos, se plantean dos escenarios: transmisión con línea de vista y sin ella. Esto con el fin de comprobar qué tan buena es la recepción de datos en distintas condiciones, haciendo una comparativa entre los datos enviados por el transmisor y lo que se muestra en la pantalla del receptor.

Para cada caso se realizan tres tomas distintas de datos, la primera es a 100 (m), la segunda a 200 (m) y la tercera a 400 (m).

**Escenario 1:** Trasmisión de la señal en línea recta en una calle vacía para asegurar línea de vista. Los resultados de la comparativa se muestran en la Tabla 3.8.

La evidencia de la recolección de datos se presenta en las Figura 3.40, Figura 3.42 y Figura 3.44 para las lecturas del transmisor, y en las Figura 3.41, Figura 3.43 y Figura 3.45 para los datos impresos en el receptor.

**Tabla 3.8** Resultados línea de vista

| Distancia (m) | Lectura del transmisor | | Lectura del receptor | | Error (%) | |
|---|---|---|---|---|---|---|
| | T (°C) | H (%) | T (°C) | H (%) | Temp. | Hum. |
| 100 | 20.6 | 61.6 | 20.6 | 61.6 | 0 | 0 |
| 200 | 18.9 | 66.4 | 18.9 | 66.4 | 0 | 0 |
| 400 | 22.2 | 58.5 | 22.2 | 58.5 | 0 | 0 |

Distancia de 100 (m):

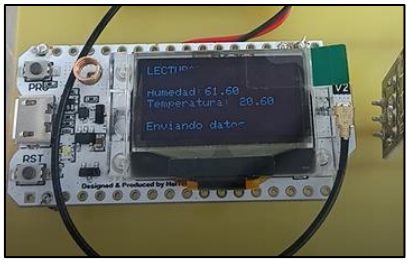


**Figura 3.40** Transmisor 100 (m)

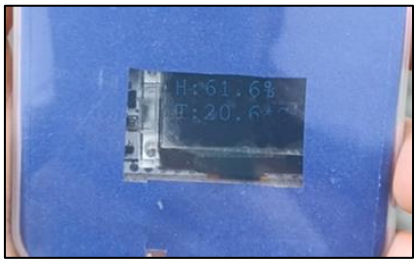


**Figura 3.41** Receptor 100 (m)

Distancia de 200 (m):

**Figura 3.42** Transmisor 200 (m)

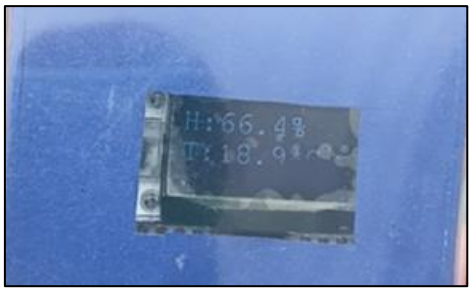


**Figura 3.43** Receptor 200 (m)

Distancia de 400 (m):

**Figura 3.44** Transmisor 400 (m)

**Figura 3.45** Receptor 400 (m)

Como se puede observar en la Tabla 3.8, cuando se tiene línea de vista entre el transmisor y el receptor no existe ningún problema de transmisión y las lecturas se reciben correctamente sin ningún tipo de retraso, tampoco se presentan impresiones de símbolos aleatorios en la pantalla ni medidas incorrectas.

**Escenario 2:** Trasmisión de la señal en la Escuela Politécnica Nacional, sin asegurar línea de vista. Los resultados de la comparativa se muestran en la Tabla 3.9.

La evidencia de la recolección de datos se presenta en las Figura 3.46, Figura 3.48 y Figura 3.50 para las lecturas del transmisor, y en las Figura 3.47, Figura 3.49 y Figura 3.51 para los datos impresos en el receptor.

**Tabla 3.9** Resultados sin línea de vista

| **Distancia (m)** | **Lectura del transmisor** | | **Lectura del receptor** | | **Error (%)** | |
|---|---|---|---|---|---|---|
| | **T (°C)** | **H (%)** | **T (°C)** | **H (%)** | **T** | **H** |
| 100 | 25.9 | 37.7 | 25.9 | 37.7 | 0 | 0 |
| 200 | 26.2 | 37.8 | 26.2 | 37.8 | 0 | 0 |
| 400 | 26.9 | 35.5 | 26.8 | 36.2 | 0.37 | 1.97 |

Distancia de 100 (m):

**Figura 3.46** Transmisor 100 (m)

**Figura 3.47** Receptor 100 (m)

Distancia de 200 (m):

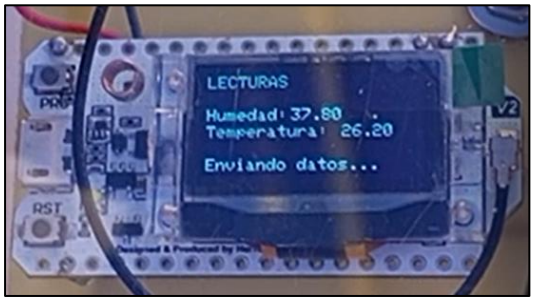


**Figura 3.48** Transmisor 200 (m)

**Figura 3.49** Receptor 200 (m)

Distancia de 400 (m):

**Figura 3.50** Transmisor 400 (m)

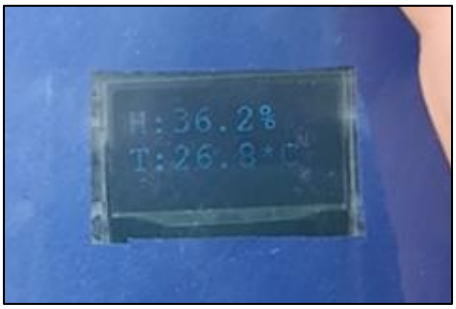


**Figura 3.51** Receptor 400 (m)

Como se observa en la Tabla 3.9, los resultados de transmisión a 100 y 200 (m) sin línea de vista son perfectos y no presentan error alguno. Por otro lado, el resultado de la transmisión a 400 (m) sí presenta fallos y cierto porcentaje de error. A pesar de que estos errores se ven mínimos, cabe aclarar que el texto impreso en la pantalla sí llegaba a ser confuso, no solo al mostrar medidas erróneas, sino al mostrar símbolos o números incompletos; esto llega a pasar incluso a 100 (m) de distancia. Debido a que la tecnología LoRa trabaja en una banda libre de frecuencias, las señales de otros sistemas pueden llegar a interferir con la transmisión de datos del prototipo. Un ejemplo de dichos errores se muestra en la Figura 3.52.

**Figura 3.52** Error de transmisión

# 4 CONCLUSIONES

- El prototipo para transmisión inalámbrica basado en tecnología LoRa desarrollado cumple con todos los requerimientos planteados. Mientras que el dispositivo transmisor tiene la capacidad de leer e imprimir datos de temperatura y humedad proporcionados por el sensor DHT22, el dispositivo receptor es capaz de recibir y mostrar dicha información en una pantalla OLED con letra legible.
- Ambos dispositivos cuentan con portabilidad a causa de la implementación de baterías de litio e interruptores para la alimentación, y también debido a su tamaño portable y estructura de protección.
- La selección del microcontrolador Heltec WiFi Lora32 para realizar la implementación del prototipo fue la adecuada, gracias a las características físicas que presenta este dispositivo se redujo el trabajo al construir la PCB para conectar el sensor e imprimir las lecturas de este; de la misma manera, debido a la compatibilidad del módulo con IDE, la programación del transmisor y receptor se realizó en un *software* accesible y sencillo de utilizar.
- El diseño del prototipo fue simple gracias a que el sensor utilizado incluye circuitería interna que proveyó la resistencia necesaria para el circuito. Para el transmisor se diseñó una placa básica con una conexión directa entre el módulo y el sensor, y una alimentación brindada por una batería. Mientras que para el receptor únicamente se proveyó alimentación con una batería, no fue necesario el uso de una PCB.
- Para la adecuada seguridad de los dispositivos, se fabricaron cajas de protección tanto para ambas partes del prototipo. La más elaborada fue la caja del módulo transmisor, para esto se diseñó y construyó una caja de acrílico capaz de proteger al sensor, y al mismo tiempo darle la ventilación adecuada para que la toma de medidas sea adecuada. La caja del módulo receptor es compacta y portable, lo que la vuelve ideal para transportar el dispositivo y cumplir con el objetivo de portabilidad.
- Las pruebas de funcionamiento del prototipo presentaron buenos resultados, tanto en la lectura de datos, como en la transmisión de estos. Se concluye que los datos recolectados de humedad y temperatura pueden verse afectados de manera situacional (clima, lugar, caja de protección, entre otros), pero que su porcentaje de error es el adecuado para asegurar que es un prototipo confiable. Por otro lado, se valida que la transmisión de datos es de largo alcance, tomando

en cuenta que a mayor distancia más notorios se vuelven los errores de comunicación, especialmente cuando no se tiene línea de vista.

- Estudiar la tecnología LoRa en IoT es beneficioso para la vida laboral y estudiantil. IoT es una tecnología en crecimiento y la demanda de expertos en esta área aumenta. La tecnología LoRa es cada vez más utilizada debido a su capacidad de conectividad de larga distancia y bajo consumo de energía. Adquirir habilidades en LoRa aumenta el valor para los empleadores y mejora las perspectivas de empleo. Para los estudiantes, proporciona una ventaja competitiva en el mercado laboral y en proyectos de investigación y desarrollo. En definitiva, estudiar la tecnología LoRa en IoT puede ser una inversión valiosa tanto para la vida laboral como estudiantil.

# 5 RECOMENDACIONES

- Debido a la forma de los pines del módulo LoRa, el contacto entre estos y los espadines no es bueno, por lo que se recomienda soldar los pines de alimentación, tierra y datos, para que el contacto y la lectura de datos sean correctos.
- Para la creación de placas PCB existen distintas técnicas para plasmar el circuito en las baquelitas; se recomienda utilizar la “técnica del marcador” cuando se tratan de circuitos sencillos ya que facilita mucho el proceso y se evitan métodos tediosos como, por ejemplo, el planchado.
- A pesar de que el prototipo cuenta con cajas de protección, se sugiere no exponerlo a situaciones extremas o que puedan afectar tanto a las cajas como a los dispositivos, ya que no son resistentes al agua o al fuego.
- Se recomienda realizar pruebas de distancia con el fin de encontrar la longitud máxima en la que el dispositivo es capaz de recibir la información. Hay que tomar en cuenta que dicha longitud se puede ver afectada por culpa de interferencias o falta de línea de vista.

# 6 REFERENCIAS BIBLIOGRÁFICAS


[1] UNIT Electronics. "Sensor de Temperatura y Humedad DHT22 con cables". UNIT Electronics. https://uelectronics.com/producto/sensor-de-temperatura-y-humedad-dht22-con-cables/ (accedido el 21 de noviembre de 2022).

[2] "LoRa Network: ¿qué es y cuáles son las aplicaciones? | Novidá". Novidá. https://www.novida.com/es/blog/lora/ (accedido el 21 de noviembre de 2022).

[3] IoT Consulting. "La tecnología LoRa y su importancia en IoT". IoT Consulting. https://iotconsulting.tech/que-es-la-tecnologia-lora-y-por-que-es-importante-para-iot/ (accedido el 21 de noviembre de 2022).

[4] "TECNOLOGÍA LORA Y LORAWAN". CATSENSORS. https://www.catsensors.com/es/lorawan/tecnologia-lora-y-lorawan (accedido el 21 de noviembre de 2022).

[5] H. Miranda. "ESPECTRO RADIOELÉCTRICO DE USO LIBRE EN ECUADOR". ISSUU. https://issuu.com/habsmiranda/docs/proyecto_final_harold_miranda (accedido el 23 de noviembre de 2022).

[6] A. Zourmand, A. L. Kun Hing, C. Wai Hung and M. AbdulRehman, "Internet of Things (IoT) using LoRa technology," 2019 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS), Selangor, Malaysia, 2019, pp. 324-330, doi: 10.1109/I2CACIS.2019.8825008.

[7] Q. Zhou, K. Zheng, L. Hou, J. Xing and R. Xu, "Design and Implementation of Open LoRa for IoT," in IEEE Access, vol. 7, pp. 100649-100657, 2019, doi: 10.1109/ACCESS.2019.2930243.

[8] "WIFI LoRa 32 (V2) (Phaseout)". Heltec Automation. https://heltec.org/project/wifi-lora-32/ (accedido el 23 de noviembre de 2022).

[9] "DHT22 – Temperature and Humidity Sensor". components101. https://components101.com/sensors/dht22-pinout-specs-datasheet (accedido el 28 de noviembre de 2022).

[10] Naylamp Mechatronics. https://naylampmechatronics.com/sensores-temperatura-y-humedad/58-sensor-de-temperatura-y-humedad-relativa-dht22-am2302.html (accedido el 28 de noviembre de 2022).

[11] X. Decuyper. "Using DHT22 Temperature/Humidity Sensor". Savjee.be. https://savjee.be/videos/programming-esp32-with-arduino/using-dht-22-temperature-humidity-sensor/ (accedido el 28 de noviembre de 2022).

[12] "LILYGO TTGO LoRa32 915Mhz V1.6.1". UNIT Electronics. https://uelectronics.com/producto/lilygo-ttgo-lora32-915mhz-v1-6-1/ (accedido el 5 de diciembre de 2022).

[13] "Sensor De Temperatura Y Humedad Dht11 Arduino". Todomicro. https://www.todomicro.com.ar/insumos/224-sensor-de-temperatura-y-humedad-dht11-arduino.html#:~:text=El%20sensor%20DHT11%20se%20caracteriza,rango%20de%200ºC%20a%2050ºC (accedido el 7 de diciembre de 2022).

[14] "PRECIO REBAJADO Sensor De Humedad Y Temperatura DHT22 Arduino Sensor De Humedad Y Temperatura DHT22 Arduino Sensor De Humedad Y Temperatura DHT22 Arduino Sensor De Humedad Y Temperatura DHT22 Arduino". TodoMicro. https://www.todomicro.com.ar/arduino/225-sensor-de-humedad-y-temperatura-dht22-arduino.html (accedido el 7 de diciembre de 2022).

[15] "MODULO DHT22 TEMPERATURA Y HUMEDAD". tiendatec. https://www.tiendatec.es/maker-zone/modulos/789-modulo-dht22-temperatura-y-humedad-8472496009805.html (accedido el 7 de diciembre de 2022).

[16] "IDE Arduino". Aprendiendo Arduino. https://aprendiendoarduino.wordpress.com/2016/12/11/ide-arduino/ (accedido el 10 de diciembre de 2022).

[17] J. Talos. "Sensor de temperatura y humedad digital DHT22". Talos Electronics. https://www.taloselectronics.com/blogs/tutoriales/sensor-de-temperatura-y-humedad-digital-dht22#:~:text=Especificaciones:,Consumo%20de%20corriente:%202.5%20mA (accedido el 12 de diciembre de 2022).

[18] D. Parmar. "Cómo calcular el porcentaje de error [+3 herramientas]". Geekflare. https://geekflare.com/es/calculate-percent-error/#:~:text=Por%20otro%20lado,%20el%20porcentaje,es%20una%20cantidad%20sin%20unidades. (accedido el 20 de febrero de 2023).

# 7 ANEXOS

La lista de los **Anexos** se muestra a continuación:

ANEXO I. Certificado de originalidad

ANEXO II. Enlaces

ANEXO III. Conjunto de datos extensos

# ANEXO I: Certificado de Originalidad

**CERTIFICADO DE ORIGINALIDAD**

Quito, D.M. 2 de marzo de 2022

De mi consideración:

Yo, ANDRÉS FERNANDO REYES CASTRO, en calidad de Director del Trabajo de Integración Curricular titulado IMPLEMENTACIÓN DE PROTOTIPOS PARA TRANSMISIÓN INALÁMBRICA BASADOS EN TECNOLOGÍA LORA elaborado por la estudiante MILENA SALOME OLMEDO LEMACHE de la carrera en TECNOLOGÍA SUPERIOR EN REDES Y TELECOMUNICACIONES, certifico que he empleado la herramienta Turnitin para la revisión de originalidad del documento escrito completo, producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del 15%.

Es todo cuanto puedo certificar en honor a la verdad, pudiendo el interesado hacer uso del presente documento para los trámites de titulación.

NOTA: Se adjunta el link del informe generado por la herramienta Turnitin.

TIC_Olmedo Lemache Milena Salomé.pdf

Atentamente,

ANDRÉS FERNANDO REYES CASTRO

Docente

Escuela de Formación de Tecnólogos

## ANEXO II: Enlaces

**Anexo II.I** Código QR de la implementación y pruebas de funcionamiento

**Enlace:** TIC_Video Explicativo_Olmedo Lemache Milena Salomé.mp4

**Enlace adicional:** https://clipchamp.com/watch/xFAOfoa2T4F

# ANEXO III: Códigos Fuente

## Módulo transmisor

```
#include <SPI.h>
#include <LoRa.h>

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 13
#define DHTTYPE DHT22

#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26

#define BAND 915E6

#define OLED_SDA 4
#define OLED_SCL 15
```

```
#define OLED_RST 16

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

DHT dht(DHTPIN, DHTTYPE);

void setup() {

  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  Wire.begin(OLED_SDA, OLED_SCL);    //Se inicia una comunicación I2C usando los pines OLED_SDA y OLED_SCL y la función Wire.begin()
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) {
    display.clearDisplay();
    display.setCursor(0,0);
    display.println(F("SSD1306 allocation failed"));
    for(;;);
  }

  display.clearDisplay();
  display.setTextColor(WHITE);
```

```
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("LORA TRANSMISOR");
  display.display();

  SPI.begin(SCK, MISO, MOSI, SS);
  LoRa.setPins(SS, RST, DIO0);

  if (!LoRa.begin(BAND)) {
    display.clearDisplay();
    display.setCursor(0,0);
    display.println("Inicio de LoRa fallido!");
    display.display();
    while (1);
  }

  display.setCursor(0,10);
  display.print("Inicializacion lista!");
  display.display();
  dht.begin();
}

void loop() {

  delay(2000);

  float h = dht.readHumidity();
```

```
  float t = dht.readTemperature();

  if(isnan(h)||isnan(t)) {
  display.clearDisplay();
  display.setCursor(0,0);
  display.println("Fallo al leer");
  display.display();
  return;
 }

  display.clearDisplay();
  display.setCursor(0,0);
  display.println("LECTURAS");

  display.setCursor(0,20);
  display.print("Humedad:");
  display.setCursor(50,20);
  display.print(h);

  display.setCursor(0,30);
  display.print("Temperatura:");
  display.setCursor(80,30);
  display.print(t);

  display.setCursor(0,50);
  display.print("Enviando datos...");
  display.display();
```

```
  LoRa.beginPacket();
  LoRa.print("H:"+String(dht.readHumidity(),1)+"%\n");
  LoRa.print("T:"+String(dht.readTemperature(),1)+"*C\n");
  LoRa.endPacket();
}
```

**Módulo receptor**

```
#include <SPI.h>
#include <LoRa.h>

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#include <Fonts/FreeMono12pt7b.h>

#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26
#define BAND 915E6

#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
```

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

String LoRaData;

void setup() {

  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }

  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("LORA RECEPTOR");
```

```
  display.display();

    SPI.begin(SCK, MISO, MOSI, SS);

  LoRa.setPins(SS, RST, DIO0);



  if (!LoRa.begin(BAND)) {
    display.clearDisplay();
    display.setCursor(0,0);
    display.println("Inicio de LoRa fallido!");
    display.display();
    while (1);
  }

  display.setCursor(0,10);
  display.print("Inicializacion lista!");
  display.display();
}

void loop() {

  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    while (LoRa.available()) {
      LoRaData = LoRa.readString();
    }
```

```
    //Impresión de los datos en la pantalla OLED
    display.clearDisplay();
    display.setCursor(0,20);
    display.setFont(&FreeMono12pt7b);
    display.print(LoRaData);
    display.display();
  }
}
```