UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DISEÑO E IMPLEMENTACIÓN DE UNA RED DE NODOS BASADO EN COMUNICACIONES DE LARGO ALCANCE (LoRa)

ALEJANDRO GÓMEZ MOLINA

2019

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DISEÑO E IMPLEMENTACIÓN DE UNA RED DE NODOS BASADO EN COMUNICACIONES DE LARGO ALCANCE (LoRa)

ALEJANDRO GÓMEZ MOLINA

TUTOR: ÁLVARO GUTIÉRREZ MARTÍN

2019

Resumen

En la actualidad existe una creciente demanda por parte de la industria que está propiciando el desarrollo de nuevas tecnologías en el sector de las telecomunicaciones. La necesidad de redes robustas, con menor consumo y mayor alcance ha desembocado en el desarrollo de nuevas tecnologías y técnicas enfocadas a las redes LPWAN y en las redes de sensores (WSN).

En este contexto, el presente Trabajo Fin de Grado se centrará en la técnica de modulación LoRa. Actualmente existen dos protocolos de comunicación que implementen esta técnica: LoRaWAN y SigFox. Un análisis de las ventajas e inconvenientes en el empleo estas tecnologías permitirá la proposición de un protocolo alternativo que utilice la modulación LoRa. Con el fin de comprobar el funcionamiento de protocolo propuesto, se diseñará e implementará un sistema completo que englobe desde el diseño hardware hasta el procesado y presentación de datos.

Mediante el análisis de los resultados obtenidos, se comprobará si el sistema cumple con las características de una red LPWAN comentándose sus ventajas e inconvenientes. Así mismo, se hará hincapié en el consumo, alcance y convergencia del sistema.

Palabras clave: LoRa, STM32, LPWAN, bajo consumo, protocolo, PCB, sistema embebido, WSN, largo alcance, tiempo real, sensores.

Abstract

At present, there is a growing demand from the industry that is increasing the development of new technologies in the telecommunications sector. The need for robust networks with lower consumption and greater range has caused the development of new technologies and techniques focused on LPWAN networks and wireless sensor networks (WSN).

In this context, the present Thesis will focus on the LoRa modulation technique. Currently, there are two communication protocols that implement this technique: LoRaWAN and SigFox. An analysis of the advantages and disadvantages of using these technologies will allow the proposal of an alternative protocol that uses LoRa modulation. In order to verify the operation of the proposed protocol, a complete system (from the hardware design to the processing and presentation of data) will be designed and implemented.

By analyzing the results obtained, it will be checked whether the systems comply with the characteristics of an LPWAN network, analyzing its advantages and disadvantages. The consumption, range and convergence of the system will be emphasized.

Keywords: LoRa, STM32, LPWAN, low power, protocol, PCB, embedded systems, WSN, long range, real time, sensors.

Agradecimientos

En primer lugar, quiero agradecer a mi tutor, Dr. Álvaro Gutiérrez Martín, por toda la ayuda prestada y por poner a mi alcance todos los medios necesarios para alcanzar los objetivos propuestos. También debo agradecer el enorme interés que ha mostrado y toda la confianza depositada en mi durante el desarrollo de este TFG.

También agradezco a todo el equipo de RBZ Embedded Logics por dejarme hacer uso de sus instalaciones, por toda la ayuda técnica que me brindaron y por todo lo que aprendí durante las horas que estuve trabajando con ellos. Por supuesto, debo mencionar a la UPM y a todos los profesores que me han formado a lo largo de la carrera. Sin su esfuerzo y dedicación docente no habría tenido las bases necesarias para desarrollar este proyecto.

Agradezco a mis compañeros del Robolabo (Blanca, Raquel, Dani...) por toda la ayuda prestada y por los largos debates sobre cosas "normales"que amenizaban las tardes de trabajo en el laboratorio. También, debo hacer una mención a mis compañeros del IEEE (Charlas, Jorge, Lucas...) por todo el apoyo técnico, teórico y psicológico prestado a los largo de toda la carrera.

Me gustaría hacer una mención especial a mi familia, por estar a mi lado en los momentos duros, por todo el amor que me han dado y por la paciencia que han tenido. Gracias por confiar y creer en mi de manera incondicional y por apoyarme en todas las decisiones que he tomado. Por último, quiero dedicar este trabajo a mi padre que, aunque ya no este con nosotros, han sido sus consejos y enseñanzas los que me han permitido llegar hasta aquí.

Índice general

Índice de figuras

Índice de tablas

Lista de Acrónimos

ADC: Analog to Digital Converter.

BLE: Bluetooth Low Energy.

BW: Bandwidth

CMRR: Common Mode Reject Ratio.

CSS: Chirp Spread Spectrum.

I^2C**:** Inter-Integrated Circuit.

IoT: Internet of Things.

LDO: Low-Dropout.

LoRa: Long Range.

LPWAN: Low Power Wide Area Network.

M2M: Machine to Machine.

NF: Noice Figure.

NFC: Near Field Comunications.

PCB: Printed Circuit Board.

SF: Spreading Factor

SMD: Surface Mount Device.

SNR: Signal Noice Ratio.

SPI: Serial Peripheral Interface.

SS: Spread Spectrum.

SWD: Serial Wire Debug.

TFG: Trabajo Fin de Grado.

UART: Universal Asynchronous Receiver-Transmitter.

UNB: Ultra Narrowband.

WSN: Wireless Sensor Network.

Capítulo 1

Introducción y Objetivos

Nos encontramos ante una auténtica revolución en el sector de la ingeniería y las telecomunicaciones: la mejora y abaratamiento de las tecnologías y redes inalámbricas están cambiando la forma en la que interactuamos con nuestro entorno. Esto, unido a la aparición de nuevas tecnologías en el procesado de datos, ha propiciado una mayor demanda por parte de la industria.

El término *IoT* (Internet of things) hace referencia a este fenómeno y consiste en la conexión de elementos cotidianos a la red. A día de hoy se puede encontrar desde bombillas hasta extensas redes de farolas interconectas. Según datos del IDC (International Data Corporation), para 2019 se prevé que la inversión mundial en *IoT* supere los 745.000 millones de dolares, lo que supone un 15,4 % más que el año pasado. Se espera que el ritmo de crecimiento continúe los próximos años llegando a superar el billón de dólares en 2022 (**?**).

Ante este panorama, se hace evidente la necesidad de innovar y desarrollar soluciones más eficientes. El creciente aumento en el número de dispositivos conectados hace necesario el desarrollo de redes y tecnologías más robustas, que soporten el gran volumen de datos actual y el esperado en años venideros.

Este Capítulo expone brevemente las tecnologías utilizadas actualmente en la industria así como sus ventajas e inconvenientes. Seguidamente, introduce los conceptos de red WSN (Wireless Sensor Network) y de red LPWAN (Low Power Wide Area Network). Por último, se centra en la tecnología LoRa (la cual se utilizará para el desarrollo de este TFG) , asi como en los objetivos planteados y la metodología aplicada.

1.1. Estado del arte

En el mercado actual podemos encontrar un amplio abanico de tecnologías de comunicación inalámbrica. Aunque en ocasiones estas tecnologías difieren mucho unas de otras, podemos caracterizarlas y compararlas mediante tres factores fundamentales: potencia, alcance y ancho de banda. Estos factores tienen fuertes interdependencias entre si, lo que hace complicado cumplirlos todos a la vez. Esta problemática se ve reflejada en la Figura **??**: si se requieren dos de los factores se tiene que renunciar al tercero.

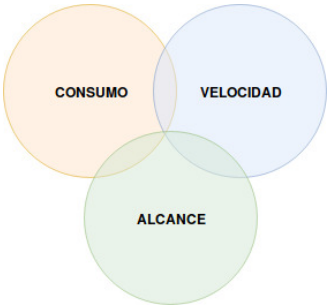


Figura 1.1: Parámetros fundamentales de un sistema inalámbrico.

Algunas de las tecnologías empleadas en *IoT* son:

- **NFC** (Near Field Communication): Estándar de comunicaciones de corto alcance inalámbricas desarrollado por Sony y NXP. Trabaja en la frecuencia de 13.56 MHz utilizando modulaciones OOK (on-off keying) y BPSK. Esta tecnología alcanza velocidades de transmisión de hasta 848 Kbps, dependiendo del entorno en el que se lleve acabo la comunicación (**?**).

- **WiFi:** Tecnología de comunicación de corto alcance inalámbrica que utiliza el estándar IEEE 802.11x. Utiliza las bandas ISM de 2.4 GHz y 5.7 GHz, utilizando un amplio número de modulaciones (varían según la banda y versión del estándar). Es uno de los estándares mas utilizados en la industria por su aplicación en redes locales (WLAN) y por las altas tasas binarias que alcanza (1.3 Gbps teóricos y hasta 400 Mbps reales) (**?**).

- **BLE** (Bluetooth Low Energy): Tecnología de comunicaciones de corto alcance, diseñada por Bluetooth Special Interest Group. Utiliza la banda ISM de 2.4 GHz y alcanza tasa binarias de hasta 1.37 Mbps (**?**). Este sistema destaca por su bajo consumo, pero tiene un alcance limitado y una tasa binaria no muy alta.

- **WiMAX** (World Interoperability for Microwave Access): Tecnología basada en el estándar IEEE 802.16, la cual puede dar servicio a redes de área metropolitana. Utiliza las bandas entre 2.3 GHz y 5.8 GHz y puede alcanzar tasas binarias de hasta 20 Mbps (**?**). Destaca por su amplio alcance (hasta 70 Km), pero requiere mucha potencia para su funcionamiento.

Además de estas, existen muchas otras tecnologías, cuyas características se resumen en la Figura **??**. En ella se hace una comparación de las diferentes tecnologías existentes en función de su alcance y consumo. Las tecnologías con las que se trabajará en este TFG son las englobadas en el cuadrante inferior derecho del esquema.

1.2. Redes WSN

Las redes inalámbricas de sensores o redes WSN (*Wireless Sensor Network* por sus siglas en inglés), se caracterizan por estar conformadas por dispositivos de bajo coste y bajo consumo (a los que nos referiremos como *nodos*) y que se utilizan para tareas de monitorización y control.

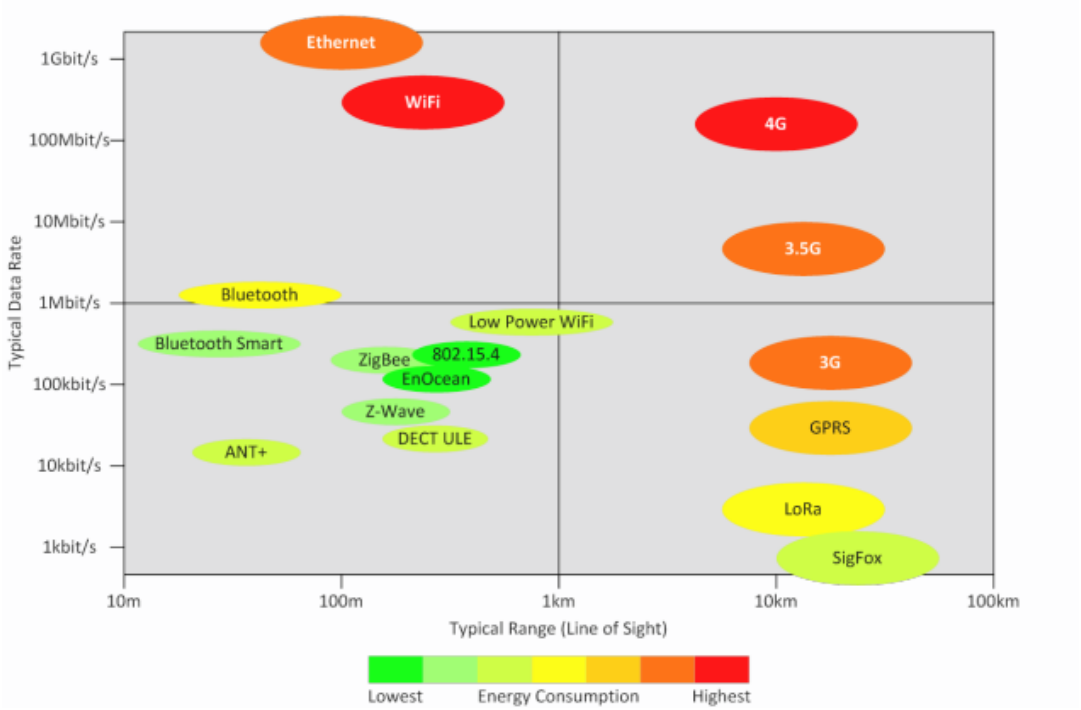


Figura 1.2: Relación de consumo, velocidad y alcance de las diferentes tecnologías (**?**).

El modo en el que se conectan e interactúan los nodos, está determinado por su topología de red, la cual define el mapa físico o lógico de los nodos en la red. Existe una extensa variedad de topologías, las cuales están resumidas en la Figura **??**. La topología de una red determinará en gran medida el consumo y coste de los nodos, la utilización de los canales de transmisión y el número máximo de nodos que pueden coexistir en una red.

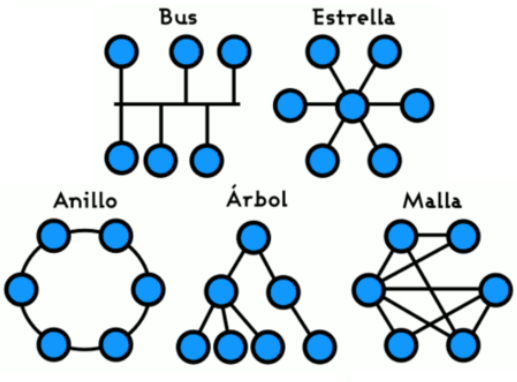


Figura 1.3: Topologías de red más usadas.

Las redes WSN suelen estar conformadas por cientos o miles de nodos distribuidos en extensas áreas geográficas. Estos factores, hacen que que la manipulación directa de cada uno de los nodos sea un proceso costoso, largo y complicado. Para evitar esta problemática, los nodos suelen tener sistemas robustos frente a fallos y autonomías que llegan a durar meses o incluso años. Todo esto es posible gracias a la utilización de diferentes técnicas de transmisión utilizadas en las redes LPWAN.

1.2.1. Redes LPWAN

Se definen como redes de bajo consumo y largo alcance (LPWAN, de sus siglas en inglés) al conjunto de redes inalámbricas caracterizadas por tener largos alcances, consumo mínimo y baja tasa binaria. Su bajo consumo posibilita implementarlo sobre dispositivos alimentados mediante baterías.

Uno de los principales inconvenientes de estas tecnologías es su baja tasa binaria, la cual hace imposible la transmisión de grandes volúmenes de datos y a su vez limita su uso a sistemas con interfaces máquina a máquina (M2M, de sus siglas en inglés). Las tasas binarias con las que trabajan las redes LPWAN rondan los 0.3 Kbps y los 50 Kbps, dependiendo principalmente de las técnicas de transmisión y los estándares utilizados.

Para conseguir estas características, las redes LPWAN utilizan diferentes técnicas de modulación como:

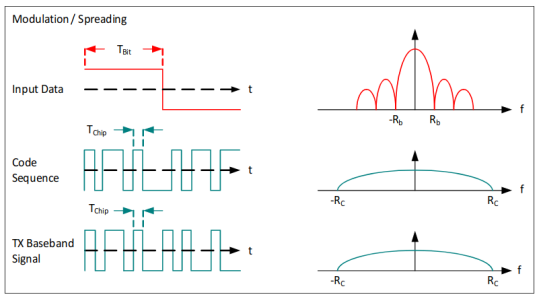
- **Ultra Narrowband (UNB):** Traducido como "banda ultra-estrecha". Su funcionamiento se basa en la utilización de canales con poco ancho de banda para trasmitir. Su reducido ancho de banda conlleva menor consumo, un uso más eficiente del espectro y un aumento drástico del número de dispositivos que pueden operar en una misma red.
- **Spread Spectrum (SS):** Se trata de una modulación de espectro ensanchado en la cual, a cada canal, se le otorga un ancho de banda mayor al estrictamente necesario para funcionar. Al aumentar el ancho de banda, aumenta la energía por bit utilizada, mejorando la relación señal a ruido (SNR, por sus siglas en inglés) en el receptor.

1.2.2. LoRa

LoRa (Long Range) es una técnica de modulación en espectro ensanchado, basada en la modulación chirp de espectro ensanchado CSS (Chirp Spread Spectrum). LoRa es una modulación propietaria, desarrollada for la empresa francesa Cycle y posteriormente adquirida por Semtech Corporation.

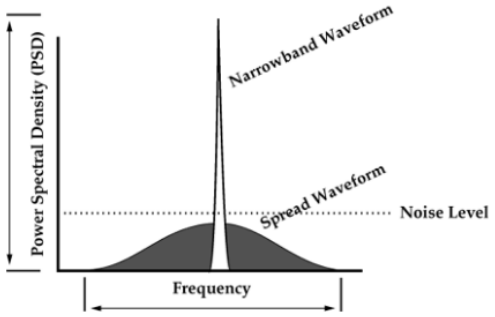
Un chirp es una señal sinusoidal cuya frecuencia varía con el tiempo. La modulación chirp de espectro ensanchado se basa en la utilización de pulsos chirp lineales para modular la señal. La utilización de estos para modular la señal hace al sistema robusto frente al desvanecimiento debido al multitrayecto, ruido pulsante en banda estrecha e interferencias debidas al efecto doppler.

Esta tecnología se caracteriza por su largo alcance, bajo consumo y bajo coste. También implementa tasas binarias variables, las cuales se configuran utilizando factores de ensanchado ortogonales (SF o Spreading Factor por sus siglas en inglés), los cuales están en escala logarítmica e indican el número de chirps por símbolo de la modulación. La utilización de estos factores permite elegir ente mejorar el alcance o la tasa binaria, utilizando un ancho de banda constante.



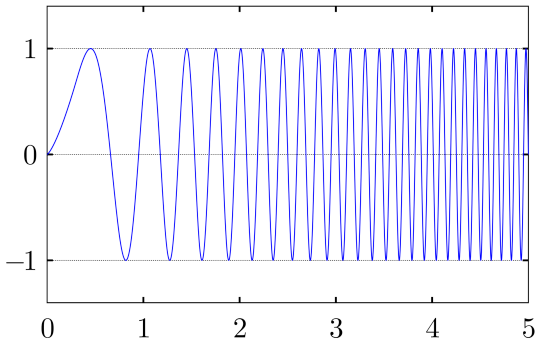
(a) Modulación, ensanchado del espectro.

(b) Demodulación, ensanchado del espectro.

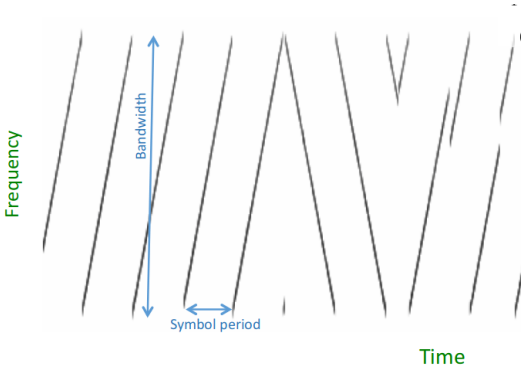


(c) Comparativa entre banda estrecha y espectro ensanchado.

Figura 1.4: Proceso de modulación y demodulación es espectro ensanchado (**?**).



(a) Representación de un chrip lineal.



(b) Representación de una señal modulada con LoRa.

Figura 1.5: Representación de un chirp y de una señal modulada con LoRa (**?**).

La relación entre la tasa binaria, el factor de ensanchado y el ancho de banda se define mediante la Ecuación **??**.Como se puede observar, al aumentar el valor de SF, disminuye la tasa binara de la modulación. Lo contrario pasa con el alcance máximo, el cual aumenta al aumentar el valor de SF.

$$R_b = SF \times \frac{1}{[\frac{2^{SF}}{BW}]}(bit/sec) \tag{1.1}$$

El SNR_{min} es la relación señal a ruido mínima que permite al receptor demodular la señal correctamente. Por otro lado, se define la sensibilidad del receptor como:

$$S_{re} = -174 + 10 * \log_{10} BW + SNR_{min} + NF \tag{1.2}$$

donde:

BW es el ancho de banda del canal,
SNR_{min} es la relación señal a ruido mínima y
NF es la Figura de ruido del receptor (6 dB)

A modo de resumen, podemos ver en la Tabla **??** un ejemplo con los valores teóricos de una implementación con un ancho de banda de 125 KHz.

SF	SNR_{min} (dB)	Sensibilidad (dBm)	Tasa binaria (Kbps)
7	-7.5	-125	6.83
8	-10	-127	3.9
9	-12.5	-130	2.19
10	-15	-132	1.22
11	-17.5	-135	0.671
12	-20	-137	0.366

Tabla 1.1: Ejemplo teórico para un canal con 125 KHz de ancho de banda.

1.3. Objetivos y motivaciones

Con la popularización de LoRa se ha extendido la utilización de LoRaWAN como protocolo de comunicación. LoRaWAN es un protocolo de red creado por *LoRa Alliance*, que busca convertirse en un estándar.

El objetivo de este TFG será implementar un protocolo de comunicación basado en LoRa, como alternativa a los ya existentes (LoRaWAN y SigFox). La principal motivación es el estudio de alternativas que impliquen un menor coste que eviten la utilización de hardware específico y que, en futuras implementaciones, soporten un mayor número de arquitecturas de red.

Seguidamente, se propondrá una diseño hardware de nodos, enfocados a bajo consumo, que utilicen transmisores LoRa. El objetivo será crear un nodo de tamaño reducido, poco consumo y que pueda ser utilizado en múltiples aplicaciones. También se diseñarán placas con sensores que implementen estos nodos y que prueben su funcionamiento. Por último, se realizará un estudio del funcionamiento del protocolo, alcance máximo y eficiencia energética de los nodos.

Cabe recalcar que la finalidad de este TFG no será crear un sistema que compita con LoRaWAN o con las redes que existen actualmente. Lo que se persigue es aplicar los conocimientos sobre redes y sistemas de comunicación, adquiridos durante el grado, para implementar un protocolo de red, diseñar un hardware que lo soporte y probar el sistema final en un entorno real.

1.4. Metodología

Con el fin de conseguir los objetivos planteados, el desarrollo se dividirá en cinco fases. El TFG está dividido en 5 capítulos ordenados con el fin de facilitar una mejor compresión del sistema y de las diferentes partes que lo componen.

En el Capítulo 2 se detalla el proceso de diseño de hardware del prototipo: se describen los componentes elegidos y sus características y se explica el proceso de diseño, fabricación y montaje final del sistema. Seguidamente, en el Capítulo 3, se detallan las especificaciones del protocolo propuesto y sus modos de funcionamiento. Tras esto, se realiza una implementación software del protocolo y se modela el comportamiento de los nodos. En este Capítulo también se describe el proceso de modificación e implementación de los diferentes drivers y herramientas utilizadas así como el proceso de prueba y depuración.

Una vez implementado el software, en el Capítulo 4 se realiza un análisis detallado del funcionamiento del prototipo. En este se detallan los resultados de las pruebas de consumo, alcance y convergencia del sistema. También se describe la plataforma diseñada para el almacenamiento, procesado y presentación de datos recogidos por los sensores.

Por último, en el Capítulo 5 se resumen los resultados obtenidos y se realiza una reflexión sobre los objetivos planteados y los conseguidos. También, se realiza una reflexión sobre los diferentes problemas surgidos durante todo el desarrollo del trabajo y se plantean posibles mejoras para futuras versiones.

Capítulo 2

Diseño hardware

2.1. Herramientas de diseño

Para la implementación hardware del sistema se ha utilizado la herramienta *Altium Designer*. Esta herramienta es un potente entorno de desarrollo que incluye todas las herramientas necesarias durante el proceso de diseño, prueba y fabricación de un prototipo hardware. Las diferentes herramientas proporcionadas por Altium se pueden ver resumidas en la Figura **??**.

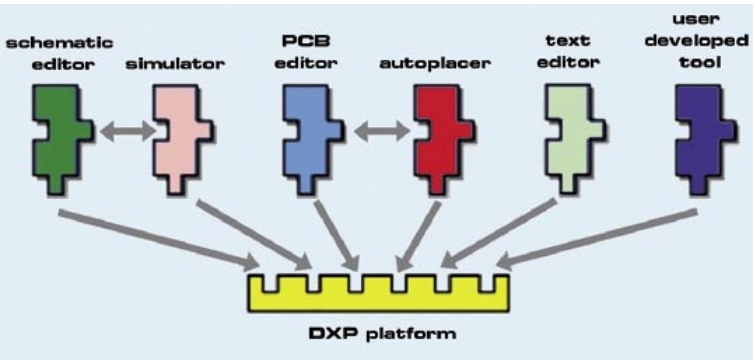


Figura 2.1: Componentes de Altium DPX (**?**).

Altium es uno de los software de diseño hardware más utilizado actualmente por la industria, y además cuenta con una gran cantidad de documentación, la cual facilita en uso y el aprendizaje. Todas estas ventajas han hecho que Altium sea la herramienta elegida para desarrollar este apartado del TFG.

Con Altium se han llevado a cabo las tareas de diseño de esquemáticos, creación de componentes y footprints, diseño y rutado de la tarjeta y la generación de los ficheros de fabricación. Aunque Altium es un software propietario con un coste elevado, para la realización de los prototipos se han utilizado licencias de prueba del software.

2.2. Elección de componentes

2.2.1. Integrado CMWX1ZZABZ

El integrado CMWX1ZZABZ es un módulo diseñado por *Murata*, que incorpora un microcontrolador STM32L072 y un tranceptor de radio SX1276 en el mismo chip.

La forma en la que se encuentran conectados ambos módulos en el chip, se resume en la Figura **??** .

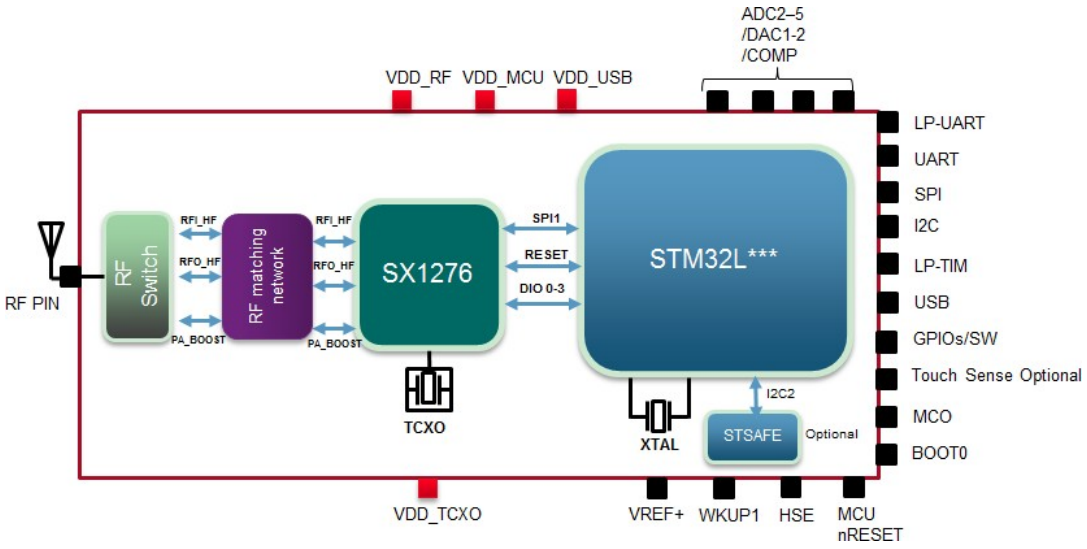


Figura 2.2: Esquema de bloques del chip CMWX1ZZABZ (**?**).

La principal ventaja de este chip es que reduce en gran medida el espacio utilizado, reduciendo el tamaño final de la tarjeta y simplificando las conexiones.

2.2.1.1. Microcontrolador STM32L072

El STM32L072, es un microcontrolador de ultra-bajo consumo, diseñado por la empresa *STMicrocontrollers*. Cuenta con una arquitectura Arm Cortex-M0+ de 32 bits funcionando a 32 MHz, además de 192 KB de memoria flash, 6 KB de memoria de datos y 20 KB de memoria RAM (**?**). Este microcontrolador tiene un amplio abanico de periféricos como SPI, I^2C, USART, USB, ADC, USB entre otros.

Lo más destacable de este microcontrolador es su ultra-bajo consumo, y sus diferentes modos de funcionamiento, los cuales se pueden ver resumidos en la Figura **??**. El bajo consumo y altas prestaciones de este chip, lo hacen ideal para este TFG.

2.2.1.2. Transceptor SX1276

El SX1276 es un transceptor de radio, diseñado por *Semtech*, preparado para trabajar con la modulación LoRa en sistemas de largo alcance y bajo consumo. También puede funcionar como modulador FSK/OOK.

Se caracteriza por trabajar en un rango de frecuencias desde 137 MHz hasta los 1020 MHz, con un *"spreading factor"* o factor de ensanchado entre 6 y 12, un ancho de banda entre 7.5 KHz y 500 KHz y una sensibilidad entre -111 dBm y -148 dBm. Con todo esto, consigue velocidades binarias efectivas ente los 18 bps y los 37.5 Kbps (**?**).

Por otro lado, el transmisor tiene un consumo típico de 10.8 mA en recepción, de entre 20 mA y 120 mA en transmisión (dependiendo de la configuración) y un consumo típico en bajo consumo (modo *sleep*) de 0.2 μA (**?**).

Low power mode	Consumption	CPU	Flash / EEPROM	RAM	DMA & Peripherals	Clock	LCD	RTC
Sleep	41 µA/MHz (Range 1)	No	ON	ON	Active	Any	Available	
	36 µA / MHz (Range 2)							
	35 µA/MHz (Range 3)							
Low power run	8.55 µA (Flash OFF, 32 kHz)	Yes	ON or OFF	ON	Active	MSI	Available	
Low power sleep	4.65 µA (peripherals OFF)	No	OFF	ON	Active	MSI	Available	
Stop with RTC	0.82 µA (1.8 V)	No	OFF	ON	Frozen	LSE, LSI	OFF	ON
	1.0 µA (3 V)							
Stop	415 nA	No	OFF	ON	Frozen	-	OFF	OFF
Standby with RTC	655 nA (3 V)	OFF	OFF	OFF	OFF	LSE	OFF	ON
	845 nA (1.8 V)							
Standby	290 nA	OFF	OFF	OFF	OFF	-	OFF	OFF

Figura 2.3: Consumo y modos de funcionamiento del microcontrolador STM32L072 (**?**).

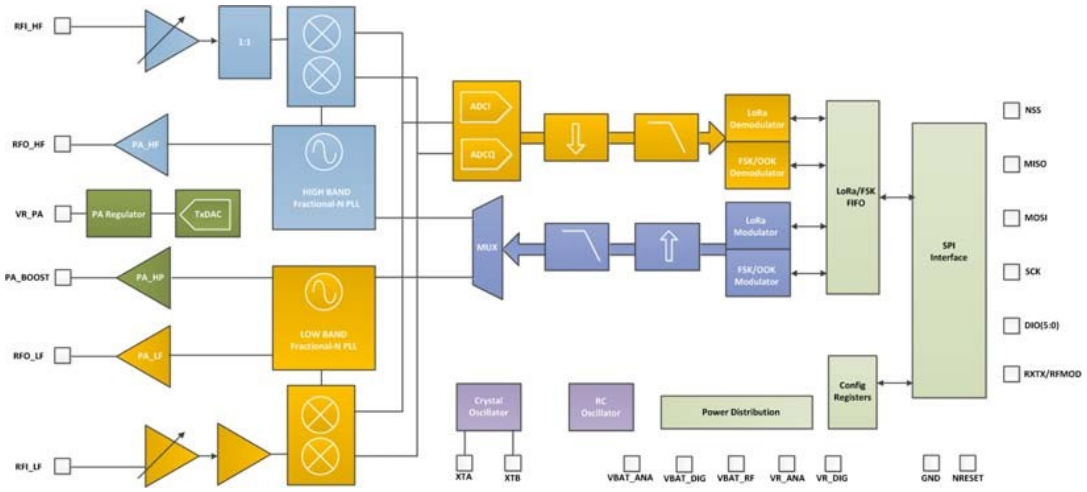


Figura 2.4: Esquema de bloques de la familia de tranceptores SX127x (**?**).

2.2.2. Conversor analógico/digital AD7194

El AD7194 es un conversor analógico/digital de bajo ruido diseñado para aplicaciones de alta precisión. Creado por *Analog*, posee 16 canales de entrada analógicos, una resolución de 24 bits, un reloj interno de 4.92 MHz y un amplificador de entrada de ganancia programable. La comunicación se hace a través de la interfaz SPI del ADC y el SPI-2 del microcontrolador.

Entre sus opciones de configuración, permite la utilización de filtros digitales programables, rechazo de las bandas de 50 Hz y 60 Hz y la utilización de sus canales en modo diferencial (en cualquier configuración) o en modo pseudo-diferencial (tomando como canal negativo GND). Otra de las características decisivas es su consumo, que varía entre los 0.85 mA y los 5.3 mA en función de la ganancia de entrada elegida (**?**).

La motivación de usar este ADC es su aplicación: la medida de termopares. En esta aplicación se hace necesario una gran precisión (las variaciones de tensión en los termopares es muy pequeña), bajo ruido de cuantificación y ganancia en la entrada.

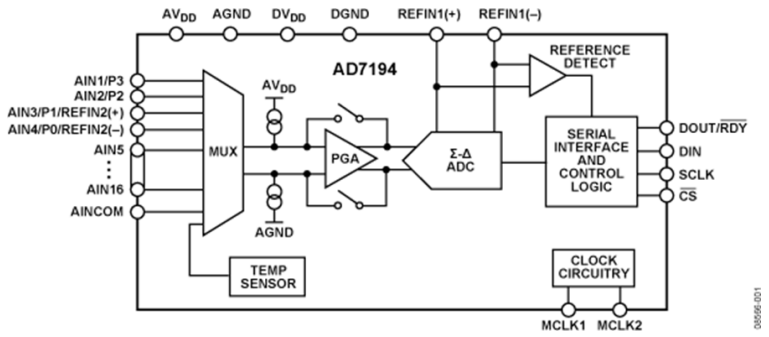


Figura 2.5: Diagrama funcional del AD7194 (**?**).

Además, al tener 16 de canales, se pueden utilizar un mayor número de termopares.

2.2.3. Sensor de temperatura MCP9808T

El MCP9808T es un sensor de temperatura digital que permite medir temperaturas entre -20 ºC y 100 ºC, con una precisión de ± 0.25 ºC. Entre su características destaca su comunicación a través de I^2C, sus diferentes modos de operación y su bajo consumo, el cual ronda los 200 μA en funcionamiento (**?**).

Su finalidad en el proyecto será aproximar la temperatura de unión de los conectores a los termopares, la cual es necesaria para realizar una correcta medida (la unión de cobre y estaño de los conectores genera una diferencia de potencial parásita que influye en las medidas). A este proceso de le llama *compensación de la unión fría.*

2.2.4. Componentes activos

Además de los ya mencionados, también se ha hecho uso de otros componentes activos, de los que podemos destacar:

- **Regulador TLV755P:** Es un regulador LDO (*Low-drowput regulator* de sus siglas en inglés) de tensión fija a 3.3 V, capaz de dar a su salida hasta 500 mA. Al ser un LDO solo necesita una diferencia de potencial de 238 mV entre sus terminales de entrada y salida, lo que nos permite alimentarlo con una batería de 3.7 V. Otro detalle importante es que su corriente de fugas en reposo es de solo 25 μA, lo que es ideal para mantener el bajo consumo.

- **Regulador AD1582BRTZ:** Este regulador es capaz de dar una tensión fija de 2.5 V con una precisión muy alta y con una corriente de fugas muy baja. En el proyecto se usará como nivel de referencia del ADC.

- **Operacional MCP6001T:** Se trata de un amplificador operacional de uso general. Funciona con una tensión de alimentación entre 1.8 V y 6 V y tiene un consumo típico de 100 μA. Se utilizará para dar alta impedancia de entrada al circuito de medida de la batería.

2.2.5. Componentes pasivos

En este apartado se engloban las resistencias, condensadores e inductancias. De manera general, se ha procurado elegir componentes con un encapsulado SMD de

tamaño 0603, el cual es lo suficiente grande para poder soldarlo manualmente sin mucha dificultad.

En cuanto a la resistencias, se han elegido de película de carbono, con una tolerancia del 1 %. Por otro lado, los condensadores son, en su mayoría, cerámicos con un dieléctrico X5R o X7R y una tolerancia $\leq 5\,\%$.

2.2.5.1. Termopares

Un termopar es un transductor de temperatura formado por la unión de dos metales diferentes. La variación de temperatura en el punto de unión de los dos metales provoca una variación proporcional de la tensión en el punto de contacto, lo que a su vez genera una corriente eléctrica a través de los conductores. A este efecto termoeléctrico se le conoce como efecto Seebeck (**?**).

La relación entre la tensión y la temperatura en el termopar se mide mediante el coeficiente de Seedbeck, el cual suele tener unidades de $\mu V/K$. Este coeficiente dependerá de los metales utilizados en la unión, de los cuales dependerán también los rangos de temperatura que será capaz de medir el termopar. Las diferentes tipos uniones se distinguen mediante letras como K (NiCr-Ni), E (NiCr-CuNi) o J (Fe-CuNi). En la Figura **??** se puede observar las variaciones de tensión frente a temperatura de los diferentes tipos de termopares.

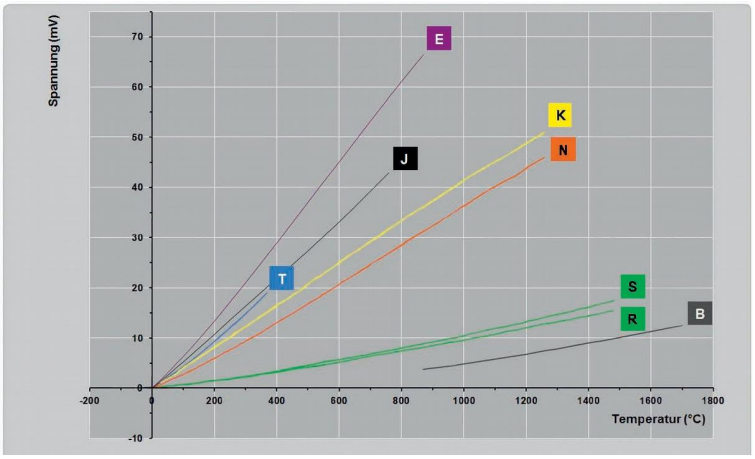


Figura 2.6: Curvas de tensión termoeléctrica (**?**).

Dado que las medidas se realizan en temperatura ambiente y que la unión cobre-estaño de los conectores forman también un termopar, la medida realizada no se corresponderá con el valor exacto de temperatura. Por ello, para obtener la temperatura absoluta es necesario realizar una compensación conocida como "compensación de la zona fría".

2.3. Diseño las tarjetas

2.3.1. Diseño del módulo

En esta parte se describirá el proceso de diseño un nodo de propósito general (sin aplicación específica definida), que contendrá lo necesario para un correcto

funcionamiento del microcontrolador y la radio. Uno de los objetivos principales será conseguir un módulo de tamaño reducido que pueda ser fácilmente montado en otro circuito impreso. Por consiguiente, se procurará reducir el tamaño de los conectores y pistas y se intentará reducir al máximo el número de componentes.

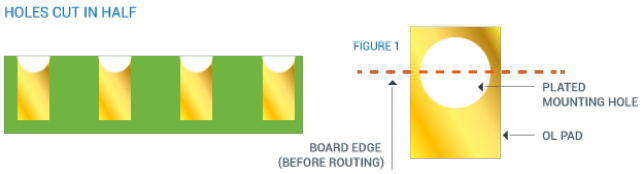


Figura 2.7: Castellated holes (**?**).

Para poder utilizar los módulos en placas de expansión, se ha optado por un montaje *placa a placa* mediante conectores de borde con *castellated holes* (Agujeros almenados, por su traducción al español, ver Figura **??**). Estos son vías (agujeros metalizados que conectan dos capas de un circuito) colocadas en los bordes de un circuito impreso, las cuales son cortadas para crear medios agujeros metalizados. Estos medios agujeros generalmente son utilizados para conectar de manera sencilla pequeños módulos como si fuesen un componente mas. La fabricación de este tipo de conectores conlleva un proceso especial, lo que eleva el precio de fabricación de los prototipos.

El módulo también cuenta con algunos elementos de depuración, como los diodos LED D1 y D2 (ver Anexo **??**), conectados a 3.3V y al GPIO PA0 del microcontrolador respectivamente. También cuenta con un botón, el cual, al ser pulsado, conecta el pin NRESET a tierra reiniciando el microcontrolador (el pin es activo a nivel bajo según la especificación del fabricante (**?**)).

Para su programación y depuración, se han añadido 5 pads, con una separación de 1.27 mm, conectados a la alimentación y a la interfaz SWD (*Serial Wire Debug* por sus siglas en inglés) del microcontrolador. Mediante estos pads y el programador STLINK, podemos crear una conexión con la herramienta de depuración GDB, la cual veremos en la Sección **??** de este documento.

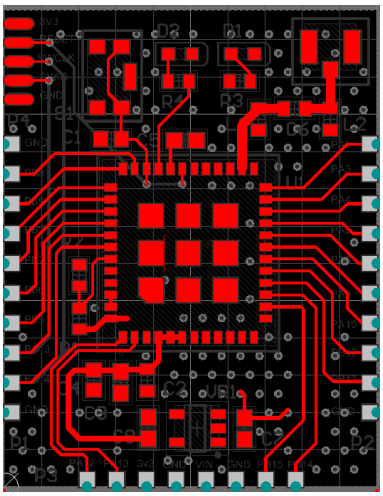
La antena va conectada al nodo mediante un conector miniatura del tipo U.LF. Los conectores U.LF trabajan en un amplio rango de frecuencias (hasta los 6 GHz), por lo que es habitual verlos en tarjetas miniPCI WiFi o en módulos GPS. Ante la imposibilidad de conseguir una buena adaptación de impedancias entre el microcontrolador y la antena (lo que causará perdidas de retorno en la línea de transmición), se ha procurado reducir la longitud de la pista que los une y se han añadido los pads para conectar una red de adaptación en el caso de que fuese necesario (componentes L1, L2 y C6 de esquemático del Anexo **??**).

2.3.1.1. Layout

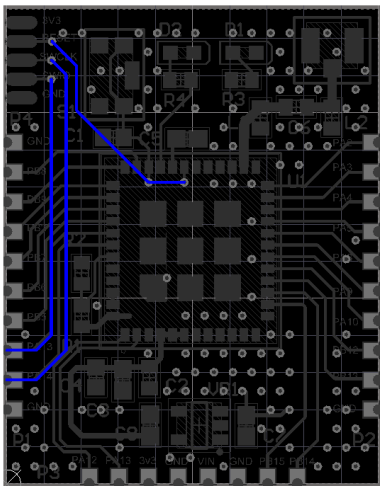
El resultado obtenido es una tarjeta de 32x25 mm^2 como la mostrada en la Figura **??**. Se ha diseñado con solo dos capas para reducir los costes y tiempo de fabricación.

Como se puede apreciar, todos lo componentes se encuentran situados en la capa superior de la PCB. Esto se ha hecho con el fin de poder usar la PCB como un módulo de montaje superficial.

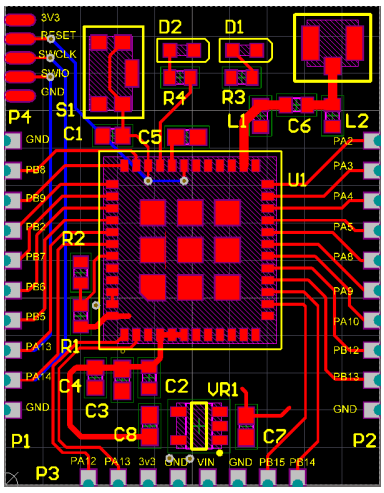
Para reducir los efectos de capacitancias parásitas y autoinductancias entre las pistas, ambas capas cuentan con un plano de masa que cubre todas las zonas libres del circuito. Ambos planos están conectados mediante un gran número de vías (también llamadas *via stitching*), la cuales ofrecen una línea de retorno rápido para las corrientes, evitan bucles en la masa y evitan que los planos funcionen como antenas. Estos planos se han ocultado en la Figura **??** para facilitar la visualización de las pistas y componentes.



(a) Elementos de la capa superior.



(b) Elementos de la capa inferior.



(c) PCB con todas las capas.

Figura 2.8: Estructura de la PCB.

2.3.2. Diseño de la tarjeta de expansión

Con el módulo funcionando, el objetivo ahora es darle una aplicación específica. Para ello, se diseñará una tarjeta de expansión con sensores con las siguientes características:

- 6 entradas para termopares.
- 3 entradas analógicas.
- Sensor de temperatura cerca de los conectores.
- Conectores enchufables con clemas.
- Sensor de batería.
- ADC de alta precisión.
- Posibilidad de cortar la alimentación.

Como ADC se ha utilizado un AD7194 como el descrito en la Sección **??**, el cual cuenta con el suficiente número de canales para soportar los 6 termopares, las 3 entradas analógicas y el sensor de la batería. Para medir la temperatura de unión se ha utilizado un sensor de temperatura MCP9808T, descrito en la Sección **??**. En los siguientes apartados se describirán de forma más detallada los esquemas utilizados para la medida de los termopares, entradas analógicas, batería y alimentación de la placa.

2.3.2.1. Medida de los termopares

Para acondicionar la entrada se ha utilizado la estructura mostrada en la Figura **??**.

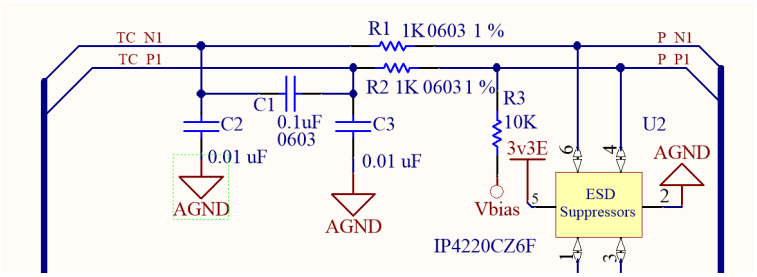


Figura 2.9: Circuito acondicionador de entrada de los termopares.

Los pares R1-C2 y R2-C3, forman dos filtros paso bajo, que filtrarán las entradas en modo común del termopar, eliminando el ruido presente en el termopar y evitando el alisasing en el ADC. Sabiendo que R1 = R2 = R y que C1 = C2 = C, se obtiene que la frecuencia de corte del filtro viene dada por la expresión:

$$f_c = \frac{1}{2 \times \pi \times R \times C} \approx 16KHz \tag{2.1}$$

Al no ser ideales, los componentes no son iguales, lo que genera una perturbación en el modo diferencial de la señal, lo que a su vez desemboca en una disminución del

ratio de rechazo al modo común o CMRR (*Common Mode Reject Ratio* por sus siglas en inglés). Como la medida de termopares requiere un alto CMRR, se ha añadido un filtro diferencial, formado por C1, R1 y R2, que compensa los efectos de los filtros en modo común. La frecuencia de corte de este filtro viene dado por la ecuación:

$$f_c = \frac{1}{2 \times \pi \times (R_1 + R_2) \times C_1} \approx 8KHz \tag{2.2}$$

2.3.2.2. Medida de entradas analógicas

Para las entradas analógicas, se ha utilizado el circuito acondicionador de la entrada de la Figura **??**.

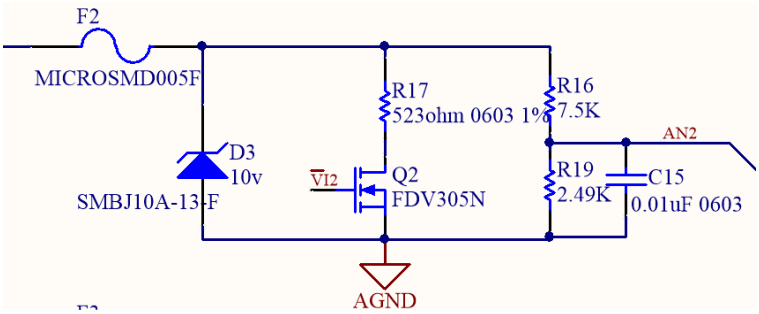


Figura 2.10: Circuito acondicionador de las entradas analógicas.

Las resistencias R16 y R19 conforman un divisor resistivo con una atenuación de 1/4 de la tensión de entrada. Teniendo en cuenta que se utiliza un regulador AD1582BRTZ a 2.5 V (como el especificado en la Sección **??**) como nivel de referencia de entrada del ADC, se consigue un margen dinámico a la entrada de 10 V.

Por otro lado, con la conmutación del transistor Q2 se consigue que la impedancia de entrada sea aproximadamente de 523 Ω, lo que permite la medida de entradas en corriente de hasta 20 mA. La expresión de la corriente a la entrada se puede expresar de la forma:

$$i_e \approx \frac{4V_m - V_{DS}}{R_{17}} \tag{2.3}$$

siendo:

V_m tensión medida en el ADC y
V_{DS} la tensión *Drain-Source* del transistor.

El circuito también cuenta con protección a su entrada: el fusible F2 limita la corriente de entrada del circuito a un máximo de 50 mA. A su vez, el diodo zener D3 limita la tensión máxima a la entrada de 10 V. En el caso de que la entrada superase los 50 mA, el fusible mantendría una corriente contante de 50 mA hasta superar el tiempo de ruptura T_r (aproximadamente 100 ms según las especificaciones del fabricante).

2.3.2.3. Media del nivel de batería

Para medir el nivel de tensión de la batería, se ha utilizado el circuito de la Figura ??.

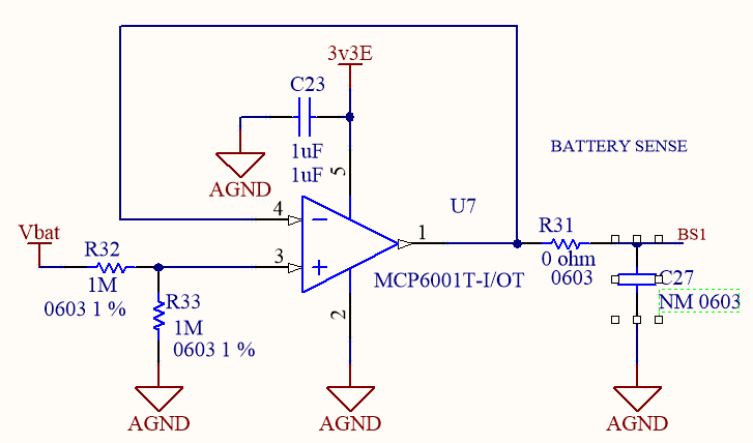


Figura 2.11: Circuito de medida de la batería.

Las resistencias R32 y R33 del circuito forman un divisor resistivo con una atenuación de 1/2. El amplificador operacional, el cual se encuentra con una configuración de seguidor de tensión, proporciona una entrada en alta impedancia (evitando así derivas de corriente). Como las resistencias están en el rango de los mega ohmios, la corriente que circula desde la batería tomará valores de micro amperios, siendo una corriente pequeña comparada con el consumo de funcionamiento.

Por último, la resistencia R31 y condensador C27 dan la posibilidad de poner un filtro paso bajo si fuese necesario. Este filtro funcionaría como filtro anti-alisaing, evitando errores en las medidas del ADC.

2.3.2.4. Alimentación

Para alimentar la tarjeta se ha utilizado un regulador TLV755P como el detallado en la Sección **??**, cuya activación se puede controlar desde microcontrolador mediante el pin de *ENABLE*. Mediante las resistencias R34 y R35 se puede elegir entre tener el regulador siempre activo o controlarlo desde el microcontrolador (ver Figura **??**.

Con el fin de eliminar el ruido de conmutación presente en la alimentación y mejorar la precisión del ADC, se ha separado la alimentación analógica de la digital mediante el filtro de la Figura **??**.

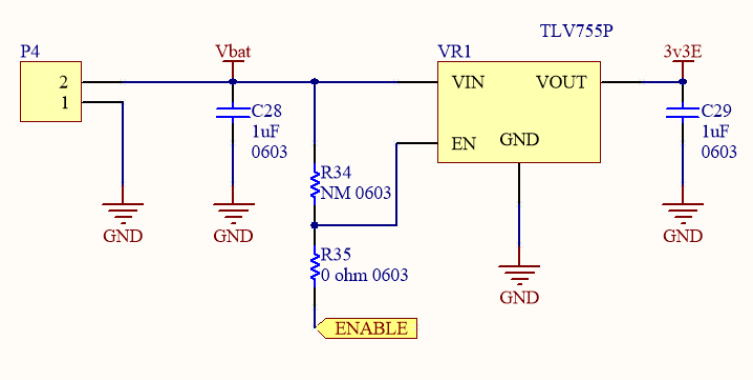


Figura 2.12: Circuito de alimentación de la tarjeta de expansión.

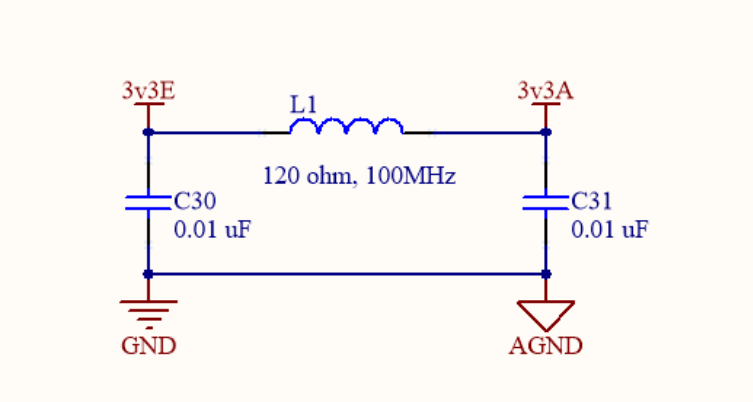
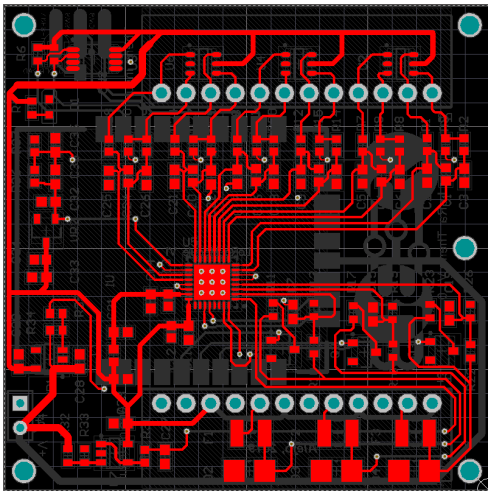


Figura 2.13: Filtro de la alimentación.

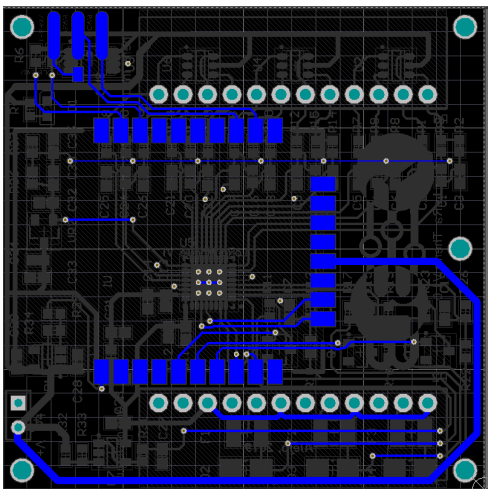
2.3.2.5. Layout

A la hora de diseñar la PCB, se han colocado los componentes en bloques funcionales, procurando acortar la longitud de las pistas. La disposición de los bloques funcionales en la PCB se ha hecho intentando reducir al máximo el espacio ocupado e intentando utilizar pistas simples y con conexión directa entre componentes. El sensor de temperatura MCP9808T se ha situado en una zona cercana a los conectores de los termopares para medir con mayor precisión la temperatura de unión.

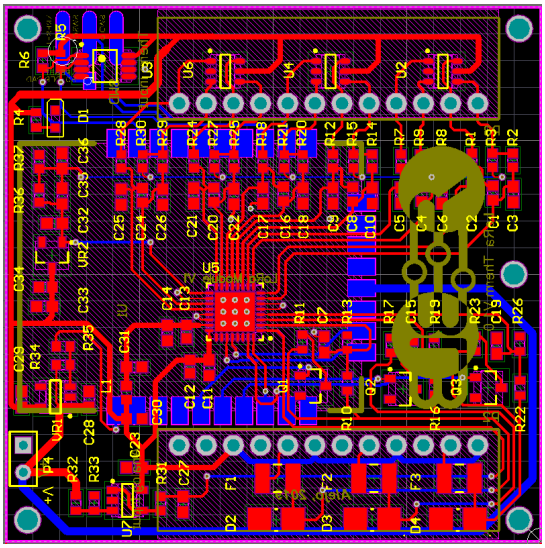
Al igual que en los nodos vistos en la Sección **??**, esta placa está diseñada a dos capas y cuenta con dos planos de masa en amabas caras del circuito, con el fin de evitar efectos parásitos en las líneas de transmisión. El resultado, es una PCB de 50x50 mm^2, cuya disposición de componentes se encuentra en la Figura **??**.



(a) Elementos de la capa superior.



(b) Elementos de la capa inferior.



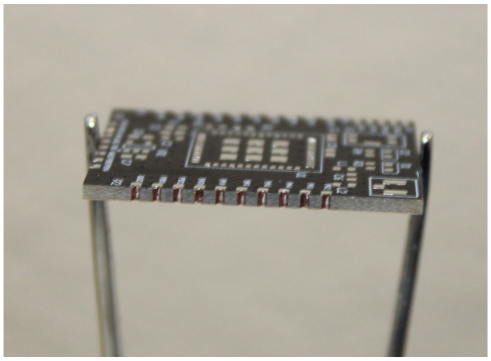
(c) PCB con todas las capas.

Figura 2.14: Estructura de la PCB.

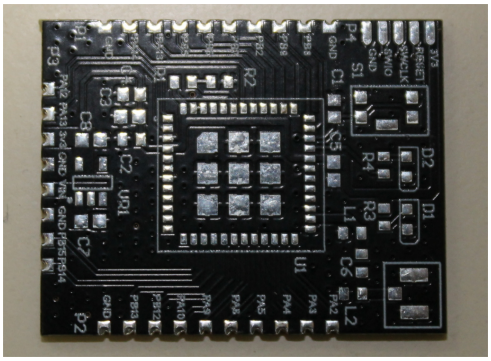
2.4. Fabricación y montaje

La fabricación de las tarjetas se llevó a cabo mediante fabricantes en china . Cada una de las placas se fabricó en empresas diferentes (Elecrow y JLCPCB) y esto se ve plasmado en la calidad y el acabado de las placas, teniendo mejores resultados en las placas fabricadas por JLCPCB. El montaje de los componentes se llevó a cabo con ayuda de la empresa RBZ embedded logics, la cual permitió el uso sus instalaciones y brindó ayuda y asesoramiento técnico durante todo el proceso.

Las placas de los módulos tuvieron algunos defectos de fabricación: como se puede ver en la Figura **??**, la deposición de estaño en los contactos no se llevó correctamente, quedando al descubierto el cobre. Esto ha supuesto serios problemas en la soldadura, debido a que la capa de óxido que se genera en el cobre impide la deposición de estaño a la hora de soldarlo. Otro defecto de fabricación se puede ver en el acabado



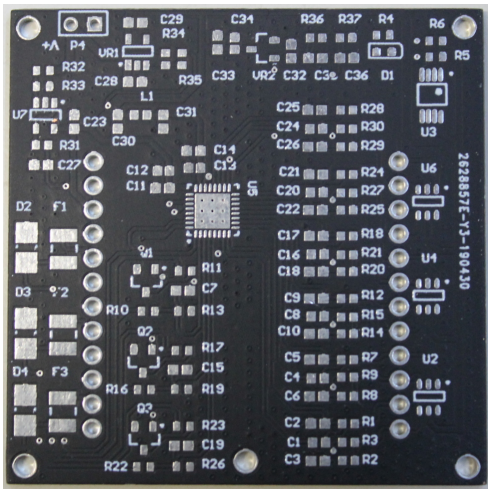
(a) *castellated holes* con defectos de fabricación.



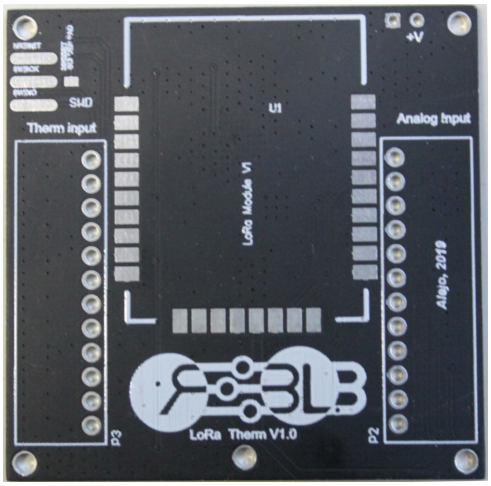
(b) Vista frontal de la PCB con terminación superficial no uniforme.

Figura 2.15: PCB de un módulo.

de superficie, el cual no es uniforme. En cambio, como se puede ver en la Figura **??**, el acabado de la placas de expansión es mucho mejor. Esto se debe a que las capacidades de fabricación del segundo fabricante son mucho mayores.



(a) Vista frontal de la tarjeta.



(b) Vista trasera de la tarjeta.

Figura 2.16: PCB de la tarjeta de expansión.

El montaje de los componentes de la tarjeta de los nodos se realizó manualmente con el soldador, a excepción del microcontrolador. Debido a la complejidad de la huella del componente (también conocida como footprint), la deposición de estaño y la alineación de componente se realizaron manualmente. Por último, se soldó el componente mediante un horno de fase de vapor.

En cuanto a la tarjeta de expansión, la soldadura se realizó totalmente con un horno de fase de vapor y con ayuda de un *stencil*. Un *stencil* es una plantilla de una PCB que permite la deposición de pasta de soldadura en todos lo pads a la vez, dosificando la cantidad justa de soldadura en cada lugar. El resultado del proceso fue el esperado, aunque algunos componentes tuvieron problemas con el efecto lápida o *tombstoning* (los componentes quedan en posición vertical debido a las tensiones internas del estaño

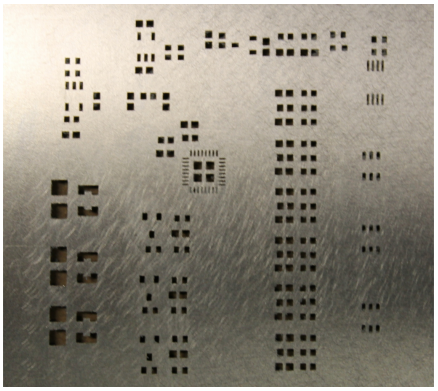
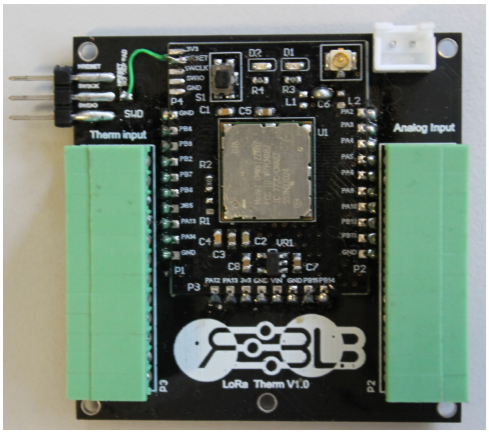
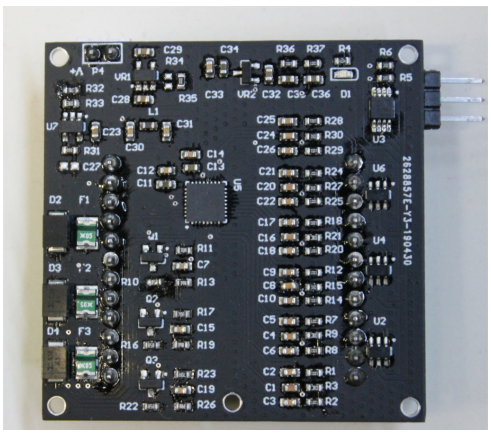


Figura 2.17: *Stencil* utilizado para el montaje.

durante el proceso de soldadura). El resultado final se puede apreciar en la Figura **??**. Se realizaron pruebas eléctricas y de software, en busca de defectos de fabricación y montaje, las cuales concluyeron satisfactoriamente.



(a) Vista frontal del prototipo.



(b) Vista lateral del prototipo.

Figura 2.18: Estructura de la PCB.

Capítulo 3

Diseño software

El protocolo de red es una de las piezas fundamentales en el desarrollo de este trabajo. En este apartado se realizará un análisis detallado de las especificaciones del protocolo, su formato de tramas y las máquinas de estados que gobiernan su funcionamiento.

Teniendo definidas las especificaciones, se describirá el proceso de implementación en la plataforma hardware diseñada: herramientas de compilación y depuración, drivers utilizados y estructura del proyecto.

Por último, se realizarán pruebas de funcionamiento, con el fin de comprobar que se cumplen las especificaciones establecidas.

3.1. Diseño del protocolo

El objetivo principal es diseñar un protocolo poco pesado, diseñado especialmente para redes de sensores y con las siguientes características:

- Tramas de tamaños variables.
- Configurable.
- Extensible a diferentes arquitecturas de red.
- Bidireccional.
- Debe funcionar sin hardware específico.
- Fácil utilización.

En esta primera versión del protocolo, se implementará una red con una topología de estrella. Para esto, se definirán dos tipos o comportamientos diferentes para los dispositivos:

- **Nodos:** Son los dispositivos que contienen sensores. Se conectan a la red administrada por un máster.
- **Master:** Se encargan de administrar los nodos de la red y de redirigir los datos que le llegan a un servidor o base de datos.

Como se verá durante el desarrollo, muchos de los formatos de tramas y características utilizados están inspirados en otros protocolos como el de Ethernet y el LoRaWAN.

3.1.1. Formato de trama

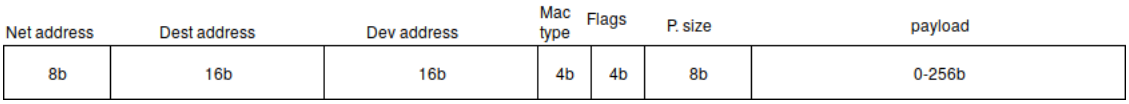


Figura 3.1: Formato de trama.

La trama enviada por el sistema está compuesta por una cabecera de 7 bytes seguida de un payload de tamaño variable entre 1 y 256 bytes. El tamaño de la carga estará limitado por la configuración de sistema. La representación gráfica de la trama se puede ver en la Figura **??**.

1. **Net address** (8b) Identificador de la red en la que se llevan acabo las transacciones. Puede tomar valores entre 0 y 254, el valor 255 está reservado para transacciones broadcast.
2. **Dest address** (16b) Identificador del nodo/ máster destino.
3. **Dev address** (16b) Identificador del nodo/ máster que realiza la transacción. Este valor puede estar definido por defecto o ser asignada por defecto por un máster.
4. **Mac type** (4b) Identificado del tipo de trama que se envía. (ver Tabla **??**).

Tipo	Identificador	Descripción
TX	0	Trama de datos
JOIN	3	Trama de conexión
ACK	1	Trama de asentimiento
NACK	2	Trama de Asentimiento negativo

Tabla 3.1: Tipos de tramas.

5. **Flags** (4b) Flags de la trama (aún no usadas).
6. **P. Size** (8b) Tamaño de la carga de la trama (expresado en bytes).
7. **Payload** (0-254) Carga de la trama. Los datos que contenga dependerán del tipo de trama (Mac type).

3.1.1.1. Trama de conexión

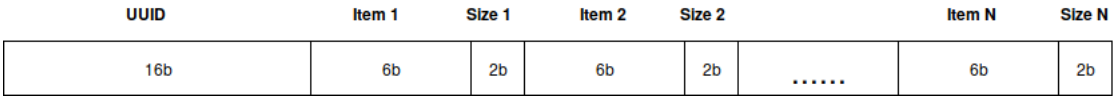


Figura 3.2: Formato de la trama de conexión.

Al iniciarse un nodo, este intentará conectarse a una red (conocida o cercana). Para hacerlo, enviará un paquete cuyo payload contendrá el identificador único de nodo, así como el formato de trama que utilizará en la transacción. El formato de la trama se puede ver en la Figura **??**, con los siguientes campos:

1. **UUID** (16b) Identificador único del nodo.
2. **Item N** (6b) Identificador del tipo de dato a enviar (ver Tabla **??**).

Tipo	Identificador	Descripción
LIGTH	0x0	Sensor lumínico
PRESSURE	0x1	Sensor de presión
HUMIDITY	0x2	Sensor de humedad
TEMPERATURE	0x3	Sensor de temperatura
.	.	.
.	.	.
.	.	.
CUSTOM 1	0x3C	Uso personalizado
CUSTOM 2	0x3D	Uso personalizado
CUSTOM 3	0x3E	Uso personalizado

Tabla 3.2: Lista de identificadores de datos.

3. **Size N** Identifica el tamaño del **Item** (ver Tabla **??**).

Identificador	Tamaño
0x0	8b
0x1	16b
0x2	24b
0x3	32b

Tabla 3.3: Identificadores de tamaño de datos.

3.1.1.2. Trama de configuración

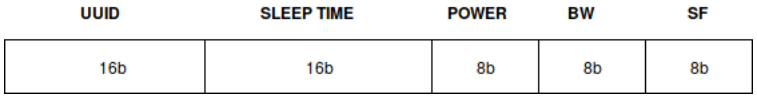


Figura 3.3: Formato de la trama de configuración.

Cuando ésta es enviada por un máster, contiene la configuración que deberá utilizar el nodo. El formato de trama se puede ver en la Figura **??**, Con los siguientes campos:

1. **UUID** (16b) Identificador único del nodo (este no cambia, solo se utiliza en la verificación).
2. **SLEEP TIME** (16b) Tiempo entre transacciones (segundos). Durante ese tiempo el nodo estará dormido.
3. **POWER** (8b) Potencia de salida de trasmisión (menor o igual a XdBm).
4. **BW** (4b) Identificador del ancho de banda del canal. Pude tomar los valores que se muestran en la Tabla **??**.

Identificador	Ancho de banda (KHz)
0x1	125
0x2	250
0x3	500

Tabla 3.4: Identificadores del ancho de banda.

5. **SF** (4b) Identificador del spreading factor del transmisor. Puede tomar los valores que se muestran en la Tabla **??**.

Identificador	SF
0x1	SF7
0X2	SF8
0x3	SF9
0X4	SF10
0x5	SF11
0X6	SF12

Tabla 3.5: Identificadores de los factores de ensanchado.

6. **RXW** (8b) valor de la ventana de recepción (en segundos).

3.1.1.3. Trama de datos

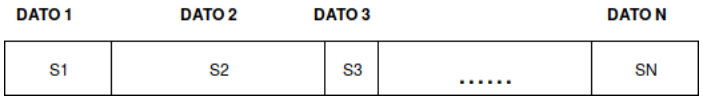


Figura 3.4: Formato de la trama de datos.

Los datos tendrán el orden y el tamaño definido en la primera conexión. El formato se puede ver en la Figura **??**.

3.1.2. Comportamiento del protocolo

En este apartado se definirá el comportamiento del sistema ante diferentes eventos.

3.1.2.1. Conexión de un nodo a la red

Cuando un nodo pretenda conectarse a un red, enviará una trama de conexión con la información referente a los sensores que tiene implementados y el tamaño del dato. Este paquete se puede enviar en modo broadcast (si se desconoce la red a la que se pretende conectar) o especificando una dirección de red.

El máster, al recibir una trama de conexión, responderá con una trama de configuración con los parámetros que usará el nodo durante su funcionamiento. Si el nodo no recibe respuesta, volverá a enviar una trama de configuración pasado el tiempo determinado en la configuración.

3.1.2.2. Envío y recepción de datos

Estando el nodo conectado a la red, podrá enviar datos al máster siguiendo el formato de datos especificado en el proceso de conexión. El nodo podrá enviar tramas con una separación temporal mínima entre ellas. El tiempo de espera entre tramas la especifica el máster en la trama de configuración (valor de *Sleep Time*).

Tras enviar un dato, el nodo abrirá un ventana de recepción, cuya duración se especifica en la trama de configuración (valor de RXW). Durante ese tiempo, estará a la espera de datos del máster. Existe la opción de hacer transacciones con asentimiento (ACK o NACK), pero no es recomendable en redes con muchos sensores debido a las limitaciones de uso del canal.

3.1.2.3. Tratamiento de errores

El sistema deberá ser robusto frente a errores. La forma en la que el sistema responderá a errores se resume en la Tabla **??**.

Nombre	descripción	solución
RX_ERROR	Paquete recibido dañado o con formato desconocido.	Se incrementará el contador de errores. **Nodo:** Si se supera el número de máximo de errores volverá al proceso de JOIN. **Máster:** descarta el paquete.
RX_NACK	El nodo máster ha recibido un paquete dañado o con formato desconocido.	Se incrementará el contador de errores. Si se supera el número de máximo volverá al proceso de JOIN.
TX_TIMEOUT	Se ha superado el tiempo máximo para enviar un paquete.	Se reconfigura la radio.
RX_TIMEOUT	Se ha superado el tiempo máximo para recibir un paquete	Si el sistema está configurado para recibir ACK, este pasará a dormir un tiempo aleatorio antes de intentar enviar un nuevo paquete.
CONF_ERROR	Existe un error el paquete de configuración	Se carga la configuración inicial en el nodo y vuelve a start.
PR_ERROR	Error al procesar la trama: Formato desconocido o datos incoherentes.	Se descarta la trama y se carga un mensaje tipo NACK en el buffer de salida.

Tabla 3.6: Tratamiento de errores.

3.2. Implementación

3.2.1. Fundamentos teóricos

3.2.1.1. Sistemas embebidos

Un sistema embebido es un sistema que a diferencia de los sistemas de propósito general, está diseñado para realizar unas pocas funciones específicas. Estas, generalmente, tienen requisitos de ejecución de tiempo real.

Estos sistemas suelen tener recursos limitados y requisitos estrictos. Por este motivo, existe un gran compromiso entre el hardware y el software (la manipulación del hardware se hace de forma directa), lo que limita su programación a lenguajes de bajo nivel.

En la mayoría de los casos, estos sistemas no pueden tener intervención humana, por lo que sus principales características deben ser:

- **Fiabilidad:** El sistema debe tener una tasa de fallos baja.
- **Robustez frente a fallo:** En caso de fallo, el sistema debe ser capaz de recuperarse de manera eficiente.
- **Disponibilidad:** El sistema debe estar disponible en el menor tiempo posible.
- **Seguridad:** En muchos casos, deben contar con funciones y protocolos que aseguren la integridad de los datos y eviten su acceso a terceros.

3.2.1.2. Sistemas operativos de tiempo real (RTOS)

Un sistema operativo de tiempo real (también conocido como RTOS por sus siglas en inglés), es un sistema operativo ligero, diseñado para sistemas embebidos con requisitos temporales estrictos. Los RTOS permiten la ejecución de código organizado en forma de tareas independientes y jerarquizadas.

Un RTOS se puede describir como un "planificador de tiempo real", ya que su función principal es organizar tareas según su prioridad y ejecutarlas garantizando tiempos de ejecución. También, administran los accesos a memoria y los recursos compartidos mediante la implementación de mutex, semáforos, colas, etc. Estos sistemas son especialmente útiles en microcontroladores con un único núcleo, los cuales solo pueden ejecutar una tarea de manera simultanea (**?**).

La utilización de un RTOS tiene grandes ventajas que facilitan el desarrollo, depuración y mejora de un sistema, entre ellas se pueden encontrar:

- **Abstracción del manejo del tiempo:** Un sistema operativo de tiempo real contiene las herramientas necesarias para controlar los tiempo de ejecución.
- **Escalabilidad y modularidad:** La organización del código en forma de tareas independientes permite añadir, quitar o modificar módulos de manera rápida y sin comprometer el funcionamiento global del sistema.
- **Reutilización del código:** La utilización de tareas independientes también permite migrar módulos a otros proyectos.

- **Control del consumo:** Los RTOS permiten tener un control estricto de los tiempo de ejecución, lo que se traduce en un mejor control del consumo del sistema.

- **Fácil depuración:** Las tareas independientes pueden ser depuradas de forma individual y aislada.

3.2.1.3. Máquinas de estados finitos (FSM)

Una máquina de estados es un modelo de diseño utilizado para modelar el comportamiento, tanto de software como de circuitos secuenciales. Estos sistemas están compuestos por estados o contextos que pueden cambiar en función de eventos o condiciones (**?**, Capítulo 7). Cada transición entre dos estados está definida mediante una condición de salto y una función de salida.

Una máquina de estados finitos permite el modelado del comportamiento de un sistemas mediante un conjunto finito de estados y un número determinado de transiciones entre ellos. Esto permite la implementación de comportamientos complejos de forma más rápida, sencilla y eficiente.

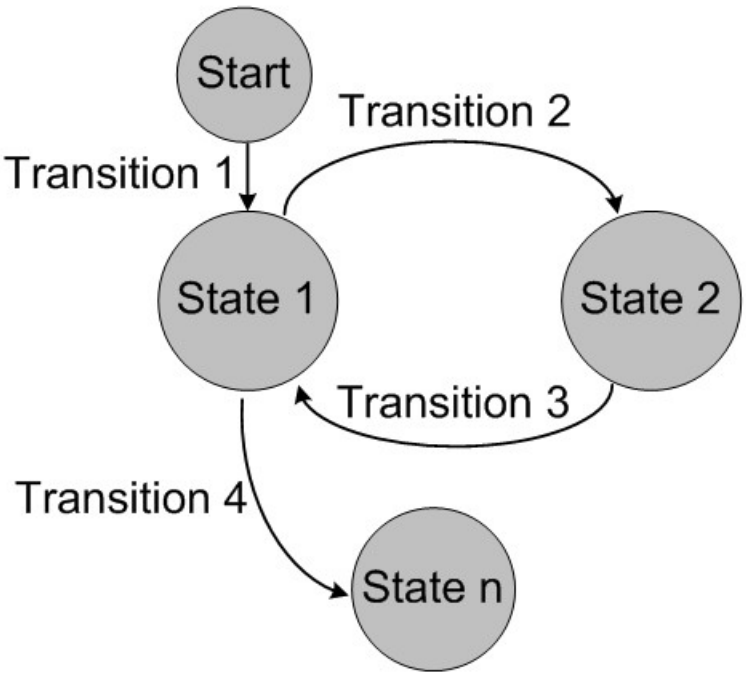


Figura 3.5: Representación de un diagrama de estados (**?**).

Estas máquinas pueden ser representadas mediante *diagramas de estados* como el de la Figura **??**. En estos, cada uno de los puntos representa un estado y las flechas definen las posibles tracciones entre estados, la condición de salto y la función de salida.

3.2.2. Descripción del entorno

3.2.2.1. Drivers, Librerías y útiles

Para la implementación del sistema, se han utilizado diferentes drivers para facilitar el proceso de implementación. Estos drivers son los encargados de controlar y gestionar los diferentes periféricos del microcontrolador y el transceptor de LoRa. Los drivers utilizados han sido diseñados por ST y Semtech para sus dispositivos, entre ellos se encuentran:

- **HAL drivers:** la *Hardware Abstraction Layer* es una iniciativa de STMicroelectronics creada con el fin de reducir el tiempo y dificultad de desarrollo de sistemas. Está compuesta por un conjunto de APIs (*Aplication Programing Interfaces* por sus siglas en inglés) que permiten inicializar, configurar y manipular los diferentes periféricos de los microcontroladores de ST. Estos drivers están accesibles a través de la red bajo una licencia de software libre BSD.

- **LoRa drivers:** diseñados por Semtech, permiten el control de sus tranceptores de LoRa. Son accesibles a través de la red y, al igual que los drivers de ST, están bajo una licencia BSD.

Como base para el proyecto, se ha utilizado uno de los ejemplos elaborados por ST para su placa de desarrollo "B-L072Z-LRWAN1", la cual se ha utilizado durante el proceso de implementación. Para utilizar este ejemplo, ha sido necesario hacer cambios el driver de transceptor, así como las secciones de configuración del hardware. La finalidad de las modificaciones era eliminar las dependencias internas con el middleware de LoRaWAN, el cual se encontraba implementado por defecto. También se reestructuró del proyecto y el sistema de compilación.

A parte de estas herramientas, también se han utilizado otras provenientes de terceros, para la implementación de las máquinas de estados y para el sistema operativo de tiempo real. Entre ellas podemos encontrar:

- **FreeRTOS:** es una implementación del kernel de un sistema operativo de tiempo real, diseñado para sistemas embebidos. Su uso de ha popularizado por la cantidad de plataformas con la que es compatible, su buena documentación y por estar bajo una licencia de software libre MIT.

- **FSM:** es una implementación de una máquina de estados basada en la máquina propuesta por J. Wang **?**, la cual es una implementación sencilla y fácilmente escalable. Esta ha sido modificada para comprobar bits de estado y no funciones en las transiciones de estados.

Para controlar los componentes de la tarjeta de expansión, se ha implementado el driver del sensor de temperatura MCP9808 y, tomando como punto de partida el driver utilizado por D. Gala Montes en su TFM (**?**), se ha reimplementado el driver del ADC AD9194 con el fin de añadir más funcionalidades y mejorar las existentes. Entre las nuevas características se pueden destacar:

- Se han extraído las funciones propias de la plataforma del código del controlador facilitando su utilización en otros sistemas con un hardware diferente.
- Se pueden configurar de forma dinámica los canales, permitiendo utilizar cualquier combinación de pares en modo diferencial.
- En la nueva versión se pueden cambiar los modos de funcionamiento, fuentes de reloj, filtros y otras opciones de muestreo de manera dinámica.
- Se han añadido funciones de calibración.
- Se ha cambiado la forma de inicialización y configuración de los canales.

3.2.2.2. Herramientas de edición, compilación y depuración

El software de este TFG se ha desarrollado completamente sobre un sistema operativo Linux pero, procurando su compatibilidad con otros sistemas operativos. Para ello, se han utilizado herramientas software libres y multiplataforma.

Para configurar el proyecto se ha utilizado *Cmake*. Esta es una herramienta libre y multiplataforma diseñada para compilar y probar software que permite generar ficheros de configuración con independencia del sistema operativo y el compilador utilizado.

Como compilador se ha utilizado el conjunto de herramientas *GCC ARM Embedded*. GCC (GNU Compiler Collection por sus siglas en inglés). Se trata de un conjunto de compiladores y librerías, creados por el proyecto GNU, que permiten la compilación de código en C, C++, Ada y Fortran. Concretamente, se ha utilizado la variante para arquitecturas ARM, la cual permite la compilación cruzada del código.

Para depurar el código, se han utilizado las herramientas GDB y STLINK. Este es un circuito integrado que permite la depuración y programación de microcontroladores STM32. Por otro lado, GDB (GNU Debugger) es un depurador de lenguajes como C y C++ que permite ejecutar instrucción a instrucción el programa y conocer, en cada momento, el estado de los registros y memoria.

3.2.3. Modelado del comportamiento de los nodos

El sistema se ha modelado mediante máquinas de estados finitos extendidas, al incluir el tiempo, cuyo comportamiento de los nodos se puede ver resumido en la Figura **??**. La función principal de los nodos será enviar datos referentes a los sensores de forma periódica al nodo maestro. El comportamiento del nodo cuenta con los siguientes 8 estados:

- **JOIN:** los nodos intentarán establecer una conexión con un máster. Para ello, enviaran un paquete de tipo JOIN como el especificado en la Sección **??**. Al llegar un paquete con estas características, el máster responderá con un otro de tipo CONFIG con la configuración del nodo, siguiendo el formato visto en la Sección **??** . Si el paquete de configuración contiene información válida, el sistema iniciará su funcionamiento habitual. Solo volverá a JOIN si se reinicia

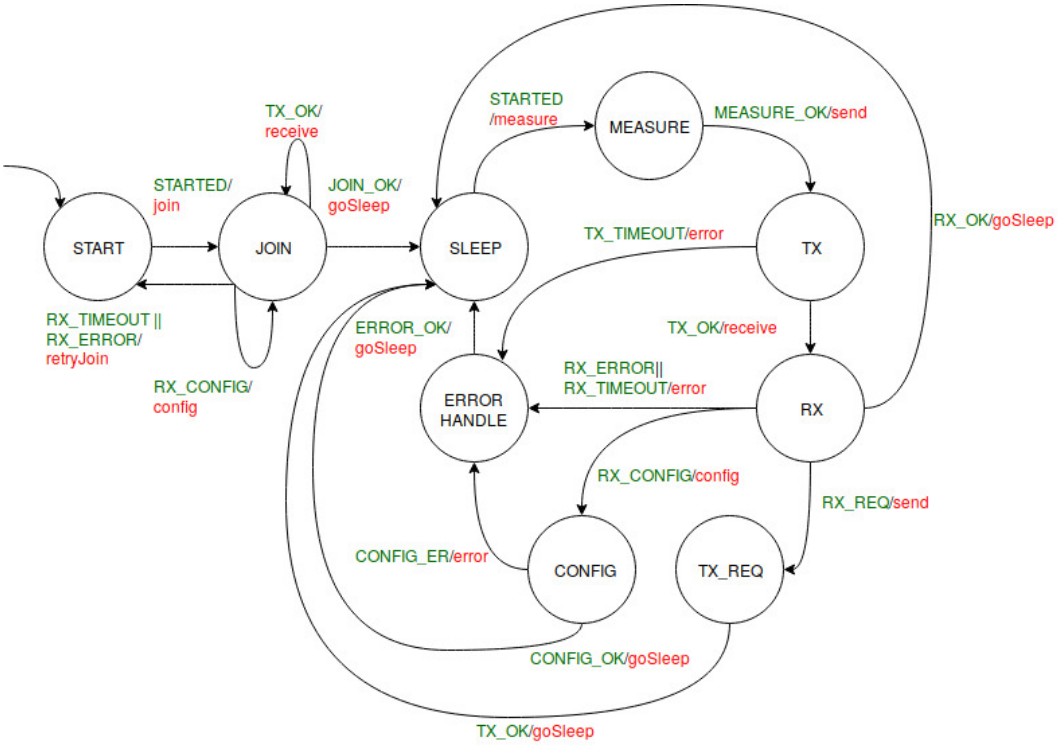


Figura 3.6: Esquema de modelado de los nodos mediante máquinas de estado.

o, si la función está habilitada, si supera el número máximo de paquetes tipo NACK definido.

- **SLEEP:** el sistema entrará en bajo consumo durante el tiempo definido. En este estado, se deshabilitarán los recursos no esenciales con el fin de minimizar el consumo del sistema.

- **MEASURE:** el sistema procesará las entradas de los sensores y cargará los datos en el buffer de salida. El formato de trama tendrá que coincidir con el especificado en proceso de JOIN, de lo contrario el nodo máster responderá con un mensaje tipo NACK.

- **TX:** El sistema enviará los elementos del buffer de salida. En caso de error, este será tratado.

- **RX:** Tras enviar los datos, se abrirá una ventana de recepción a la espera de un paquete de datos. En caso de error, este será tratado.

- **CONFIG:** Si el paquete recibido es de tipo configuración (ver Sección **??**), esta se cargará en el sistema y será utilizada en la siguiente transacción.

- **TX_REQ:** Si el paquete recibido es del tipo request info, se cargará la información del sistema en un paquete tipo configuración y se enviará al nodo máster.

- **ERROR_HANDLE:** Si ocurriese algún error durante la ejecución de alguno de los procesos, este se capturará y se intentará buscarle una solución. La forma en la que se tratarán los errores se encuentra especificada en la Sección **??**.

3.2.4. Modelado del comportamiento del nodo maestro

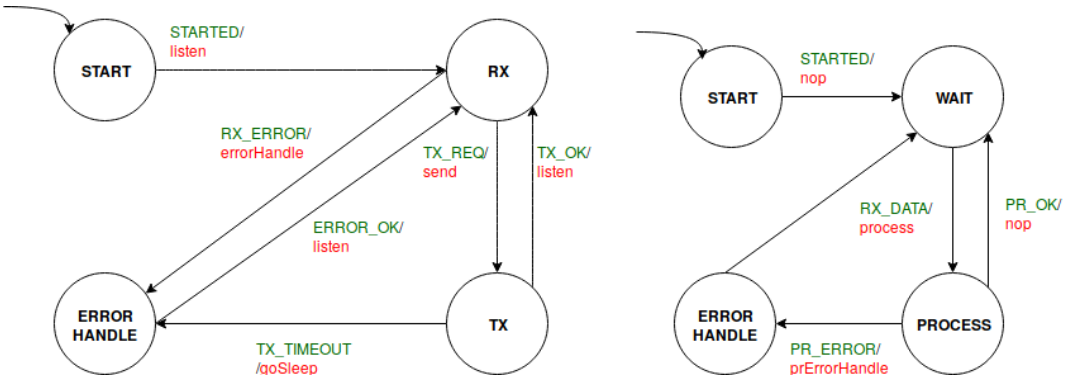


Figura 3.7: Modelado del comportamiento mediantes maquinas de estado extendidas.

A diferencia de los nodos vistos en la Sección **??**, el comportamiento de los nodos maestros se ha modelado mediante dos máquinas de estados. La principal motivación de este cambio será separar la funciones de recepción y transmisión de paquetes de las funciones de procesado de datos. De esta forma, evitamos la pérdida de datos procedentes de los nodos. Todo esto es posible gracias a las posibilidades que nos ofrece FreeRTOS y las máquinas de estados finitos extendidas. Los estados de la máquina de recepción de datos son los siguientes:

- **RX:** este es el estado por defecto del receptor. Se encuentra siempre escuchando el canal a la espera de datos. Cuando recibe datos los ubica en una cola circular y avisa a la segunda máquina de la presencia de datos.
- **TX:** cuando la segunda máquina notifica la existencia de datos en el buffer de salida, esta envía los datos y vuelve a su estado por defecto.
- **ERROR_HANDLE:** recupera al sistema de los errores notificados por otros estados.

Por otro lado, los estados asociados al procesado de datos son los siguientes:

- **WAIT:** Se mantiene a la espera de datos.
- **PROCESS:** Cuando existen datos encolados, los procesa y genera un respuesta.
- **ERROR_HANDLE:** Recupera el sistema de un error.

Los estados y las transiciones asociadas de cada máquina se pueden ver resumidos en la Figura **??**.

3.2.5. Estructura del código

La forma en que se estructura el proyecto se puede ver resumido en la Figura **??**. Se puede dividir la estructura en tres bloques principales: *Nodes* (contiene la implementación de los nodos), *Master* (contiene la implementación del nodo maestro) y *global* (contiene los drivers, middlewares y herramientas utilizados de manera

general). Dentro de la carpeta *include* de los bloque nodo y maestro, se encuentran los ficheros de configuración del sistema (*system_config.h* y *FreeRTOSConfig.h*). Por otro lado, en la carpeta *source* se encuentran la implementación del protocolo.

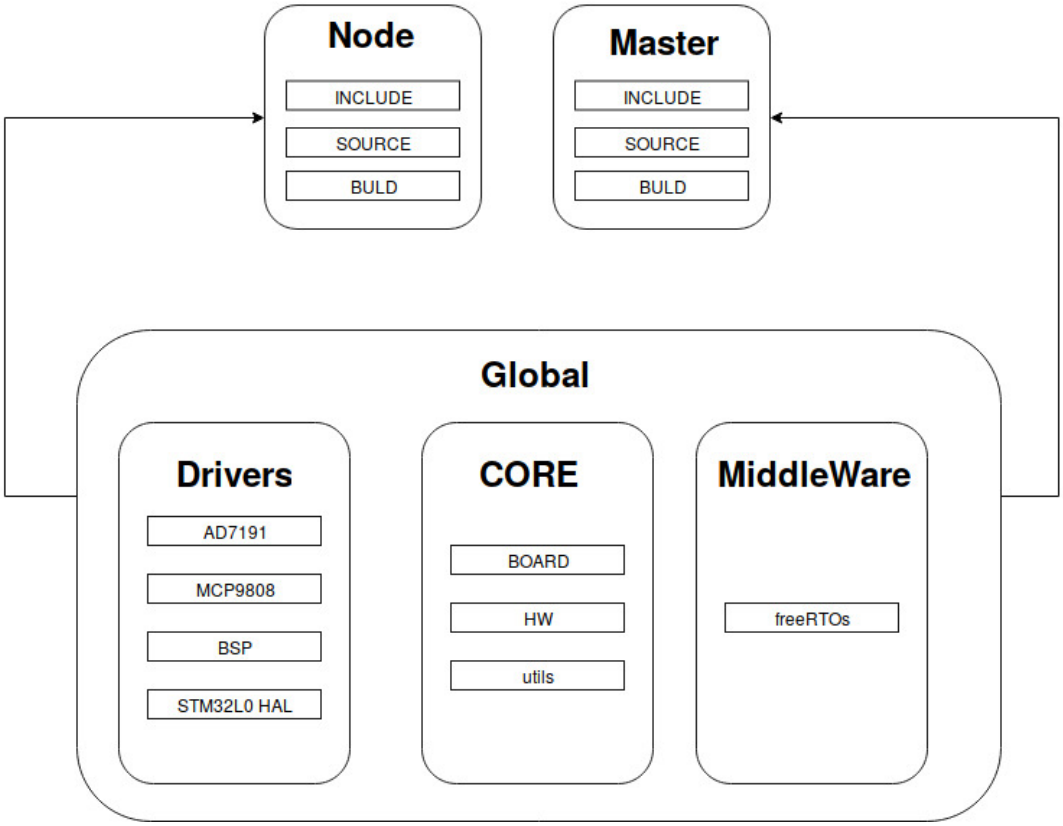


Figura 3.8: Estructura del proyecto.

La carpeta global contiene todos los drivers y herramientas utilizados para la implementación del protocolo, los cuales están divididos en tres subgrupos:

- **Drivers**: contiene todos los drivers que controlan el hardware del sistema. La carpeta *BSP* contiene los drivers de la radio adaptados al tipo de microcontrolador usado, la carpeta *STM32L0_HAL* contiene los drivers de capa de abstracción de hardware proporcionada por Semtech y las carpetas *AD7194* y *MCP9808* contienen los drivers del ADC y del sensor de temperatura.
- **MiddleWare:** contiene el código de freeRTOS.
- **Core:** contiene la configuración global del proyecto y las herramientas utilizadas. En la carpeta *BOARD* se encuentra la configuración del hardware utilizada, en la carpeta *HW* están las funciones de control del hardware y en la carpeta *Utils* las herramientas utilizadas (implementación de colas, depuración por trazas, maquinas de estado, entre otros).

Capítulo 4

Validación y pruebas del sistema

En este capítulo se detalla el proceso de validación del sistema final. Para validar un sistema electrónico es necesario realizar una batería de pruebas que incluye inspecciones visuales, pruebas eléctricas y funcionales del sistema con el objetivo de garantizar el correcto funcionamiento del prototipo.

El proceso de inspección visual consiste en la búsqueda de errores de fabricación y montaje apreciables a simple vista. Entre los posibles errores podemos encontrar aquellos en las huellas de los componentes, problemas con el acabado de los terminales, componentes faltantes o mal soldados, entre otros. Este proceso, así como los errores encontrados se detallan el la Sección **??** de fabricación y montaje de las tarjetas. Las pruebas restantes y sus resultados se detallan en las siguientes apartados de este capítulo.

4.1. Pruebas eléctricas

Para que el sistema funcione correctamente, es necesario que todos los componentes y sus conexiones lo hagan de esta manera. Para comprobar que esto es así, se deben realizar pruebas que descarten cortocircuitos, líneas cortadas, y que garanticen que todos los componentes estén correctamente conectados.

La primera prueba consiste en comprobar las pistas de las tarjetas. Aunque es frecuente que el fabricante las realice antes de su envío, es conveniente realizar una segunda comprobación con el fin de evitar la pérdida de componentes. Con ayuda de un polímetro digital, se ha probado la conducción de las pistas y el aislamiento entre ellas. También se han comprobado las conexiones a masa y las pistas de alimentación.

Tras el montaje de los componentes y su inspección visual, se ha comprobado que todos estuvieran bien conectados y que funcionaran correctamente. Para ello, se comprobaron los valores de tensión a la entrada y salida de los reguladores, así como a la entrada de los componentes. De igual manera se ha buscado comprobar que los componentes pasivos tuvieran el valor y el comportamiento esperado. Estas pruebas encontraron que uno de los fusibles no funcionaba correctamente, lo cual facilitó su reemplazo.

El resultado de las pruebas fue satisfactorio al no encontrarse problemas en las líneas, ni defectos que no se pudieran corregir de manera rápida. Además, los pequeños problemas encontrados evitaron malas interpretaciones de los resultados de las pruebas funcionales.

4.2. Pruebas funcionales

Estas pruebas se centran en comprobar el funcionamiento de los componentes y del software que los controla. Estas servirán para determinar posibles fallos eléctricos no detectados en el apartado anterior y encontrar fallos de software.

4.2.1. Pruebas de conexión con los componentes

En este apartado se centrará en comprobar el funcionamiento del ADC AD7194 y del sensor de temperatura MCP9808. Para realizar las pruebas se ha empleado un programa sencillo que utiliza los drivers implementados para intentar leer algunos de los registros de los chips. Las pruebas del AD7194 se basan en intentar leer el valor del sensor de temperatura interno y uno de sus canales de entrada. Por otro lado, para probar el MCP9808 se intentará leer la temperatura y el identificador del chip.

```
//--------------> begin System task functions <--------------//
void main_task(void* param){

    static float temp1 = 0, temp2 = 0, analog = 0;
    uint16_t manId;

    //init hardware
    therm_init();
    ExtADC_Init();
    ExtADC_Reset();

    //Config channel
    extADC_t config;

    config.gain = EXTADC_GAIN_1;            //Input gain
    config.mode = PSEUDO;                   //Input mode
    config.ExtConfigOptions = UNIPOLAR;     //Input type

    ExtADC_ConfigChannel(BATTERYC, &config);

    while(1){
        temp1 = ExtADC_ReadTempSensor();            //Internal Temp Sensor
        analog = ExtADC_ReadVoltageInput(CHANNEL_1);//Analog input (connected to Vcc)

        manId = MCP9808_GetManId();                 //Manufacturer ID
        MCP9808_GetTemp(&temp2);                    //temp Value

        vTaskDelay(1000/portTICK_RATE_MS);

    }

}
```

Figura 4.1: Código utilizado en las pruebas.

Para realizar estas pruebas, se hará uso de la herramienta GDB para conocer el valor de las variables utilizadas. El código usado se puede ver en la Figura **??** y el resultado de las pruebas puede verse resumido en la Figura **??**. Gracias a estas se pudo encontrar algunos errores en las configuración del ADC que impedían la

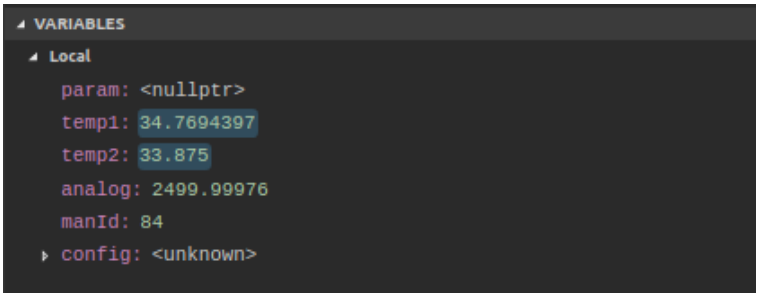


Figura 4.2: Resultado de las pruebas.

correcta lectura de los canales. Entre los errores encontrados, destacan fallos en la configuración de los periféricos y problemas en la configuración de los registros. Estas pruebas también permitió encontrar un fallo en el bus I^2C (las líneas de datos y reloj estaban cruzadas). El problema se solucionó cortando las pistas y volviéndolas a unir con un cable.

4.2.2. Pruebas de transmisión/recepción

Las transmisión y la recepción son las partes más importantes del sistema. Por ello, en este apartado se buscará la comprobación del correcto funcionamiento del driver, de la radio y del protocolo. Para ello, se pondrá en funcionamiento con algunos de sus sensores y se analizarán los datos recibidos por el nodo maestro.

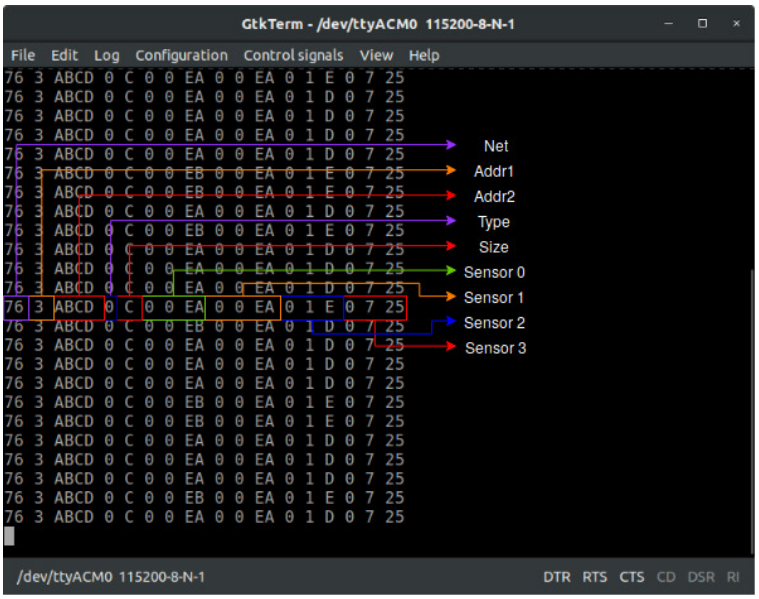


Figura 4.3: Prueba de recepción de datos.

El resultado se puede ver en la Figura **??**. El formato de trama recibido coincide con el formato de trama de datos especificado en las secciones **??** y **??** para un nodo con cuatro sensores de 24 bits, coincidiendo los valores con los esperados.

4.2.3. Pruebas de las entradas de termopares

Para probar el funcionamiento de las entradas en modo diferencial, se inicializaron las 6 entradas de termopares. Después se conectó un termopar tipo T en cada una de las entradas y se hizo variar la temperatura del sensor para ver su respuesta.

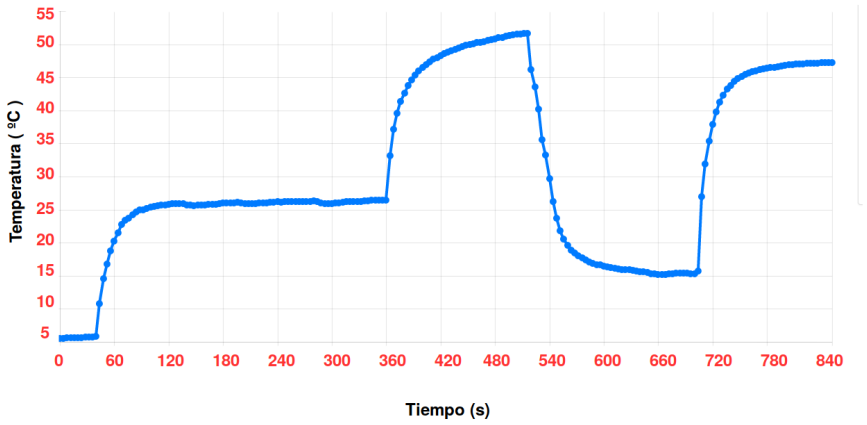


Figura 4.4: Prueba de funcionamiento de los termopares.

El resultado de una de las pruebas se puede ver en la Figura **??**. Para esta prueba se enfrió el sensor a una temperatura cercana a los 0ºC; tras esto, se dejó el sensor a temperatura ambiente hasta estabilizarse a unos 25ºC. Seguidamente se aplicó un foco caliente que la elevó hasta aproximadamente 50ºC y se volvió a enfriar hasta unos 15ºC. El resultado de las pruebas fue satisfactorio para todas las entradas de los termopares.

4.2.4. Pruebas de las entradas analógicas

Para probar estas entradas, se configuraron sus respectivos canales en modo bipolar. Tras esto, se hizo variar su tensión de entrada entre 0 y 10 V en pasos de 1 V. En cada paso, se dejaba la tensión estable durante algunos segundo para permitir al nodo tomar y enviar varias muestras por cada valor de tensión.

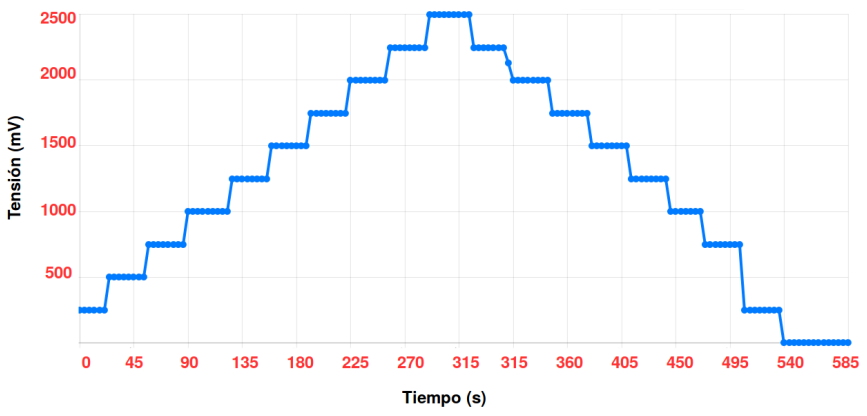


Figura 4.5: Prueba de funcionamiento de las entradas analógicas.

El resultado de la prueba se puede apreciar en la Figura **??**. Tal y como se especifica en la Sección **??**, la tensión de entrada está atenuada por un factor de $\frac{1}{4}$, por lo que lo que la gráfica varía entre 0 y 2.5 V.

4.3. Validación del sistema completo

4.3.1. Plataforma de prueba y presentación de los datos

Con el fin de comprobar el funcionamiento del dispositivo en un tiempo amplio, se ha diseñado una plataforma de almacenamiento y presentación de datos. La plataforma consta de dos partes fundamentales.

Por un lado, se ha diseñado un servidor en python encargado de administrar los nodos, las direcciones y los datos. En el momento de una nueva conexión, el servidor será el encargado de asignar una dirección y los parámetros de configuración del nodo. Durante la recepción de datos, los almacena según el formato establecido durante la conexión y, de ser necesario, comunica al maestro los datos que debe transmitir al nodo. Este servidor se comunica con el nodo maestro mediante puerto serie. Los datos recibidos se almacenan en una base de datos SQL, con la estructura de la Figura **??**.

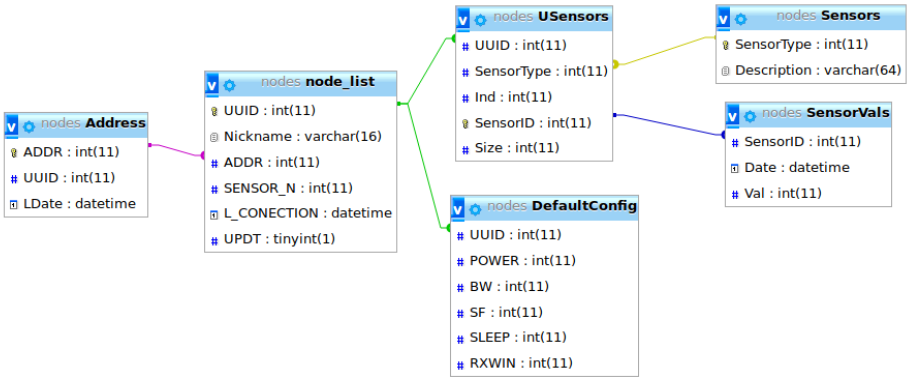


Figura 4.6: Estructura de datos del servidor.

Para presentar los datos y poder configurar los parámetros de los nodos, se ha diseñado una página web en *php*. De esta manera, se pueden consultar los datos procedentes de los sensores de cada nodo. Además, se pueden ver las características de los datos (máximo, mínimo y media, entre otros) y se pueden modificar los parámetros de nodos (apodo, tiempo entre muestras, ancho de banda, etc). Cuando los parámetros de un nodo han cambiado, el servidor espera a recibir una nueva trama de dicho nodo para notificar los nuevos parámetros (mediante una trama de tipo *config* como la explicada en la Sección **??**).

4.3.2. Pruebas de conexión de los nodos a la red

Hasta el momento, solo se ha probado el proceso de conexión y trasmisión de un único nodo a la vez. El principal objetivo de este apartado será su comprobación. Con ayuda del servidor especificado en la Sección **??**, se comprobará si todos los nodos

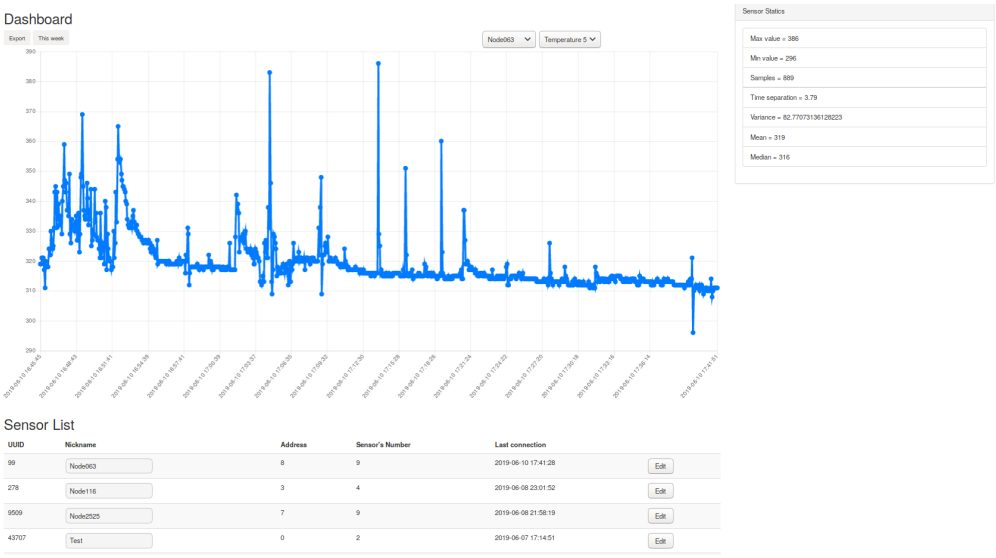


Figura 4.7: Plataforma web de presentación de datos.

logran conectarse y transmitir los datos de sus sensores. El resultado de la prueba puede, finalmente, considerarse satisfactorio.

4.3.3. Pruebas de alcance

Para comprobar el alcance que tienen los nodos, se han realizado pruebas desplazando su posición a diferentes distancias. Para estas pruebas se utilizaron los parámetros de la Tabla **??**.

Parámetro	Valor
RX Window	1 s
Sleep Time	1 s
Spreading	7
TX Power	14 dBm
Bandwidth	500KHz

Tabla 4.1: Parámetros utilizados para las pruebas de alcance.

Como se puede observar en la Figura **??**, el alcance máximo obtenido fue de 260 m. Este alcance es escaso si se compara con los alcances teóricos del transmisor. Sin embargo, hay que tener en cuenta que las condiciones en que se realizaron las pruebas no eran ideales:

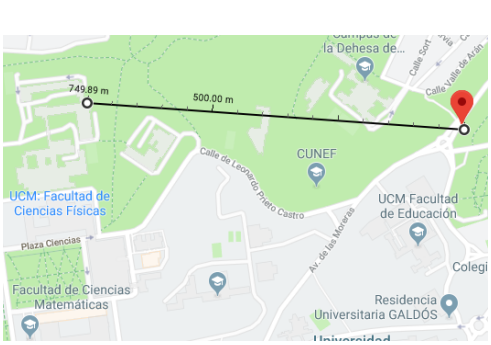
- Las pruebas se realizaron en una zona con un terreno con cambios de altura bruscos.
- En la zona había múltiples edificios de gran altura que provocaban pérdidas por difracción.
- El nodo no transmitía a máxima potencia.
- No había visión directa entre el transmisor y el receptor.

- Se probó en una zona urbana ruidosa.

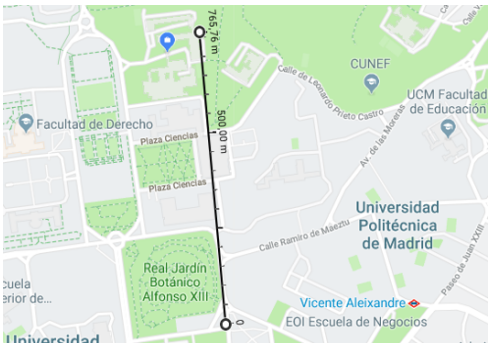
Figura 4.8: Distancia máxima alcanzada por un nodo.

El motivo de realizar las pruebas en estas condiciones es conocer el funcionamiento en un entorno real. Se consideró que comprobar los nodos en condiciones óptimas no aportarían suficiente información sobre el comportamiento del sistema en condiciones adversas.

Tras estas pruebas, se realizaron unas segundas pruebas de alcance en mejores condiciones. Para realizarlas, se utilizaron los parámetros de la Tabla **??**. Como se puede ver en la Figura **??**, con estos parámetros hemos conseguido un alcance máximo de aproximadamente 800 m. Hay que tener en cuenta que aunque mejoramos los parámetros de los nodos, siguen habiendo grandes pérdidas por difracción y por ruido.



(a) Alcance en dirección este.



(b) Alcance en dirección sur.

Figura 4.9: Pruebas de alcance con parámetros mejorados.

Parámetro	Valor
RX Window	1 s
Sleep Time	1 s
Spreading	12
TX Power	20 dBm
Bandwidth	500KHz

Tabla 4.2: Parámetros mejorados utilizados para las pruebas de alcance.

4.3.4. Pruebas de autonomía

Una de las principales características de una red LPWAN es su bajo consumo. En este apartado se analizará el consumo del prototipo y se hará una estimación de su autonomía. Para aproximar el consumo, se ha realizado medidas de la corriente suministrada y el tiempo durante las principales tareas del sistema. Esta tarea se llevará a cabo con ayuda de un amperímetro y el depurador GDB. Los parámetros de configuración utilizados durante las pruebas son los resumido en la Tabla **??**.

Parámetro	Valor
RX Window	1 s
Sleep Time	10 m
Spreading	7
TX Power	17 dBm
Bandwidth	500KHz

Tabla 4.3: Parámetros utilizados para el análisis de consumo

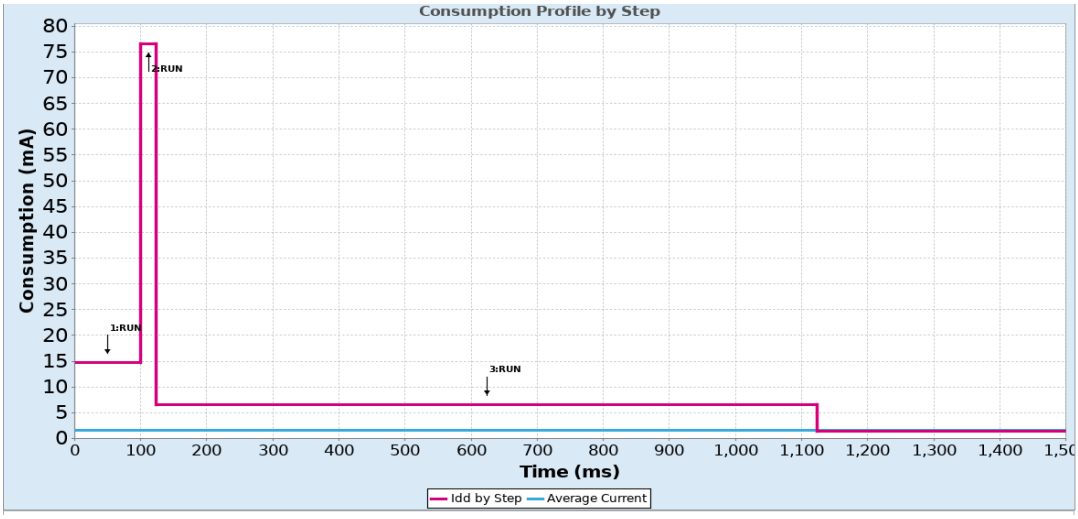


Figura 4.10: Análisis de consumo del sistema

En la Figura **??** se puede ver el consumo del sistema en los tramos más altos y duraderos (durante los procesos de medida, envío, recepción y espera). El primer tramo se corresponde con el proceso de medida de las entradas, el cual tiene

una duración aproximada de 50 ms por cada entrada medida y un consumo de aproximadamente 15.5 mA. Tras este proceso, el sistema envía los datos elevando el consumo a unos 75 mA durante 24 ms. Como se describió en la Sección **??**, tras enviar los datos el sistema abre una ventana de recepción, que en este caso tiene una duración de 1s y un consumo de 6.5 mA. Por último el sistema entra en modo parada (STOP), reduciendo su consumo a 1.5 mA.

Analizando la Figura **??**, se puede observar que tanto el consumo a lo largo de los procesos de medida, como el envío, son los esperados según las especificaciones de los fabricantes (**?** y **?**). Sin embargo, el consumo en modo parada (STOP) es un orden de magnitud más alto que el especificado por el fabricante (ver Figura **??**). Para intentar determinar el origen del consumo tan elevado, se ha aislado la alimentación de la tarjeta de expansión y se han retirado la mayoría de los integrados (probando el consumo en cada extracción).Tras numerosas pruebas eléctricas y de software no se ha podido determinar con total claridad la razón de este consumo tan elevado. Sin embargo, se sospecha que el microcontrolador no está entrando en modo parada (se queda en algún estado intermedio) o que los reguladores no están funcionando según lo esperado.

Con los resultados obtenidos, se ha podido determinar el consumo medio del sistema y la su autonomía aproximada. En la Tabla **??** se pueden ver los resultados obtenidos. Estos cálculos se han realizado tomando como referencia una batería de litio con una celda a 3.7 V y con una capacidad de 2000 mAh.

	Valor
Consumo Medio	1.51 mA
Autonomía en funcionamiento	$\approx 900h$
Autonomía en reposo	$\approx 930h$

Tabla 4.4: Análisis de autonomía del sistema

Capítulo 5

Conclusiones y líneas futuras

5.1. Conclusiones

Tal y como se planteó en el primer capítulo de este documento, los principales objetivos planteados eran diseñar e implementar un protocolo de comunicación basado en la modulación LoRa, diseñar un hardware que lo soporte y crear una pequeña red de nodos con el fin de probar el funcionamiento del sistema.

Con el fin de conseguir los objetivos planteados, se realizó un análisis de la tecnología LoRa y de los protocolos que actualmente lo implementan y se propuso un sistema alternativo. Para probar su funcionamiento se diseñó un hardware con sensores de temperatura que implementase dicho protocolo. Se ha hecho énfasis en el tamaño y el consumo del prototipo final. Al sistema final se le realizaron numerosas pruebas eléctricas y funcionales para validar el sistema.

Durante el proceso de validación del sistema, se encontraron algunos errores de diseño hardware, los cuales se pudieron solucionar rápidamente sin necesidad de rediseñar ni refabricar las tarjetas. De la misma manera, a lo largo de todo el proceso, se fueron encontrando y solucionando diversos problemas en el código implementado.

Con los resultados obtenidos en el capítulo anterior, podemos concluir que el sistema cumple con las especificaciones planteadas, sin embargo, los resultados son mejorables. El consumo en reposo, aunque es bajo, sigue siendo demasiado elevado, lo que aleja a nuestro sistema de los valores ideales. Por otro lado, aunque el alcance de nuestros nodos es elevado, no se acerca a los alcances kilométricos que puede alcanzar un sistema con esta tecnología. Frente a esto podemos recalcar que las pruebas se realizaron en zonas y parámetros de configuración con condiciones no muy óptimas.

Tras probar una red conformada por 5 nodos trabajando de manera simultánea, el sistema completo de los nodos ha demostrado ser robusto y que puede coexistir con más sistemas. Esto demuestra que el protocolo podría tener un uso potencial en redes de sensores reales.

En general, del resultado obtenido podemos concluir que es un sistema sencillo que permite desplegar rápidamente una red de nodos con un número variable de sensores. La sencillez y versatilidad del protocolo hace posible su implementación en numerosas

aplicaciones enfocadas al *Internet of Things* (IoT) y las redes de sensores (WSN). A su vez, el diseño modular del hardware acelera el proceso de diseño y fabricación y aumenta el número de aplicaciones potenciales. En esta línea, el sistema podría ser utilizado en tareas de monitorización y control, sobre todo en zonas remotas con difícil acceso a la red eléctrica.

A término personal, el desarrollo de este proyecto me ha permitido profundizar en el diseño hardware y software de sistemas empotrados enfocado al bajo consumo. También ha aumentado mi comprensión de la redes de comunicación y los sistemas que las soportan. Uno de los detalles más destacables es que he podido aprender todo el proceso de diseño, fabricación y validación de un sistema electrónico de la mano de personas que se dedican profesionalmente a ello.

5.2. Líneas futuras

Durante el desarrollo de este TFG han surgido un gran numero de ideas para mejorar el desarrollo del sistema. De la misma manera, de los errores de diseño cometidos han surgido ideas para mejorar el prototipo. Entre ellas podemos destacar:

- Implementar arquitecturas de red más complejas. Se ha procurado que el diseño del protocolo permita la futura implementación de redes *mesh*.
- Diseñar un servidor más versátil que permita más opciones de configuración y que mejore la gestión de los datos recibidos.
- Crear un sistema máster que utilice más de un nodo en recepción. De esta manera se podrían hacer cambios de configuración adaptativos y dedicar un canal exclusivo para paquetes de configuración y señalización.
- Implementar una capa de seguridad en el sistema (actualmente los datos van sin encriptar).
- Añadir opciones de geolocalización a los nodos.
- Con ayuda de los paquetes de señalización, implementar un sistema de detección de fallos en la red que no haga un uso excesivo del canal.
- Implementar el funcionamiento de los nodos en entornos con más de un máster funcionando. Este es un escenario muy probable en un despliegue real de estos sensores.
- Mejorar el diseño de las tarjetas, poniendo más atención a la impedancia de las pistas (sobre todo las de las antenas).
- Mejorar el consumo del sistema. Esto sería posible mejorando el diseño del hardware y mejorando la eficiencia del software.

ANEXOS

Anexo A

Aspectos éticos, económicos, sociales y ambientales

La creciente evolución de la tecnología está cambiando la forma que las personas ven e interactúan con el medio cotidiano. El estar conectado se está convirtiendo en una necesidad cada vez más extendida y demandada por la población. En este contexto, el desarrollo e imputación de nuevas tecnologías que satisfagan esta necesidad, tendrá un importante impacto directo sobre la sociedad.

A.1. Impacto social

El presente TFG brinda la posibilidad de crear redes de sensores de bajo consumo, de forma rápida y sencilla. Este posibilitará el despliegue de redes de monitorización y control en zonas urbanas y rurales. Al no depender de las redes de comunicación actuales y estar enfocada al bajo consumo, permitiría su implementación en zonas aisladas.

Todos estos factores darían a la población un mayor grado de conocimiento y control del medio en el que viven, lo que a su vez podría desembocar en un aumento de su calidad de vida. Por otro lado, su implementación en ambientes agrícolas e industriales, podría mejorar la productividad, disminuyendo los costes de producción (lo que repercutiría de forma directa en el consumidor).

A.2. Impacto económico

La propuesta presentada en este proyecto permite la implantación de redes inalámbricas privadas e independientes. El impacto económico más directo es la posibilidad de prescindir (en gran medida) de las redes móviles actuales y de los costes de acceso a estas. La conexión de cientos de nodos a las redes móviles actuales supondría un coste muy elevado que repercutiría de manera directa sobre los usuarios.

Por otro lado, este sistema podría mejorar la eficiencia y la productividad de los diferentes sectores económicos. Una mayor monitorización de las diferentes fases de producción, facilitaría la detección y prevención de fallos lo que a su vez, evitaría pérdidas económicas en materias primas y paradas en el proceso productivo. En la

misma línea, el despliegue de sistemas de control, permitiría reaccionar de manera rápida ante imprevistos y realizar operaciones de manera remota.

A.3. Impacto medioambiental

La implantación de estas tecnologías, podría mejorar la eficiencia energética. Su utilización en *Smart Cities* podría reducir el consumo eléctrico mediante la creación de redes de farolas y semáforos, los cuales podrían regular su funcionamiento en función de la hora y número de personas. También podrían favorecer de manera indirecta a la reducción de la contaminación ya que los datos de los sensores evitarían desplazamientos innecesarios.

Cabe desatracar, que la producción de estos dispositivos genera una huella medioambiental debido a la extracción de metales raros (utilizados para la creación de los componentes), los químicos necesarios para la producción, el litio utilizado en las baterías y al enorme coste energético que conlleva todo el proceso de fabricación. Sin embargo, el diseño modular del dispositivo, permite su reutilización en diferentes proyectos, aumentando su vida útil y disminuyendo la huella medioambiental que genera.

A.4. Conclusiones

Pese a la huella medioambiental de fabricación del sistema, los potenciales beneficios son altos. El presente poyecto podría contribuir al desarrollo de las *Smart Cities* y del IoT. Tal y como se ha visto en los apartados anteriores, su utilización en estos campos repercutiría positivamente en la sociedad, mejorando y facilitando la interconexión de elementos cotidianos a la red ofreciendo a los usuarios un mayor control sobre su entorno.

Además, podría reportar beneficios económicos al optimizar los procesos productivos en una gran parte de los sectores económicos. A su vez, la optimización de estos procesos, reduciría la huella ambiental que generan. La flexibilidad del sistema permite su reutilización en diferentes aplicaciones.

Anexo B

Presupuesto económico

	Horas	precio hora	total
COSTES DE PERSONAL	500	16.00 €	8,000.00 €

COSTES DE RECURSOS MATERIALES

	Precio	Tiempo de uso (meses)	Depreciación (años)	Total
Ordenador personal	1,500.00 €	8	4	250.00 €
Material de laboratorio	3,000.00 €	8	4	500.00 €
Licencias de software	3,000.00 €	8	1	2,000.00 €
TOTAL				2,750.00 €

Gastos generales (costes indirectos)	15.00 %	1,612.50 €
Beneficio industrial	6.00 %	741.75 €

MATERIAL FUNGIBLE

	Total
Pasta de soldadura	10.00 €
Estaño	3.00 €
Flux	3.00 €
Cable SW30	2.00 €
Latiguillos	4.00 €
Plástico PLA	3.00 €
Componentes	250.00 €
Placas	36.00 €
Baterías	30.00 €
Antenas	15.00 €

SUBTOTAL		13,460.25 €
IVA APLICABLE	21.00 %	2,826.65 €

TOTAL DEL PRESUPUESTO	**16,286.90 €**

Tabla B.1: Resumen del presupuesto del proyecto

Anexo C

Esquemáticos de los nodos

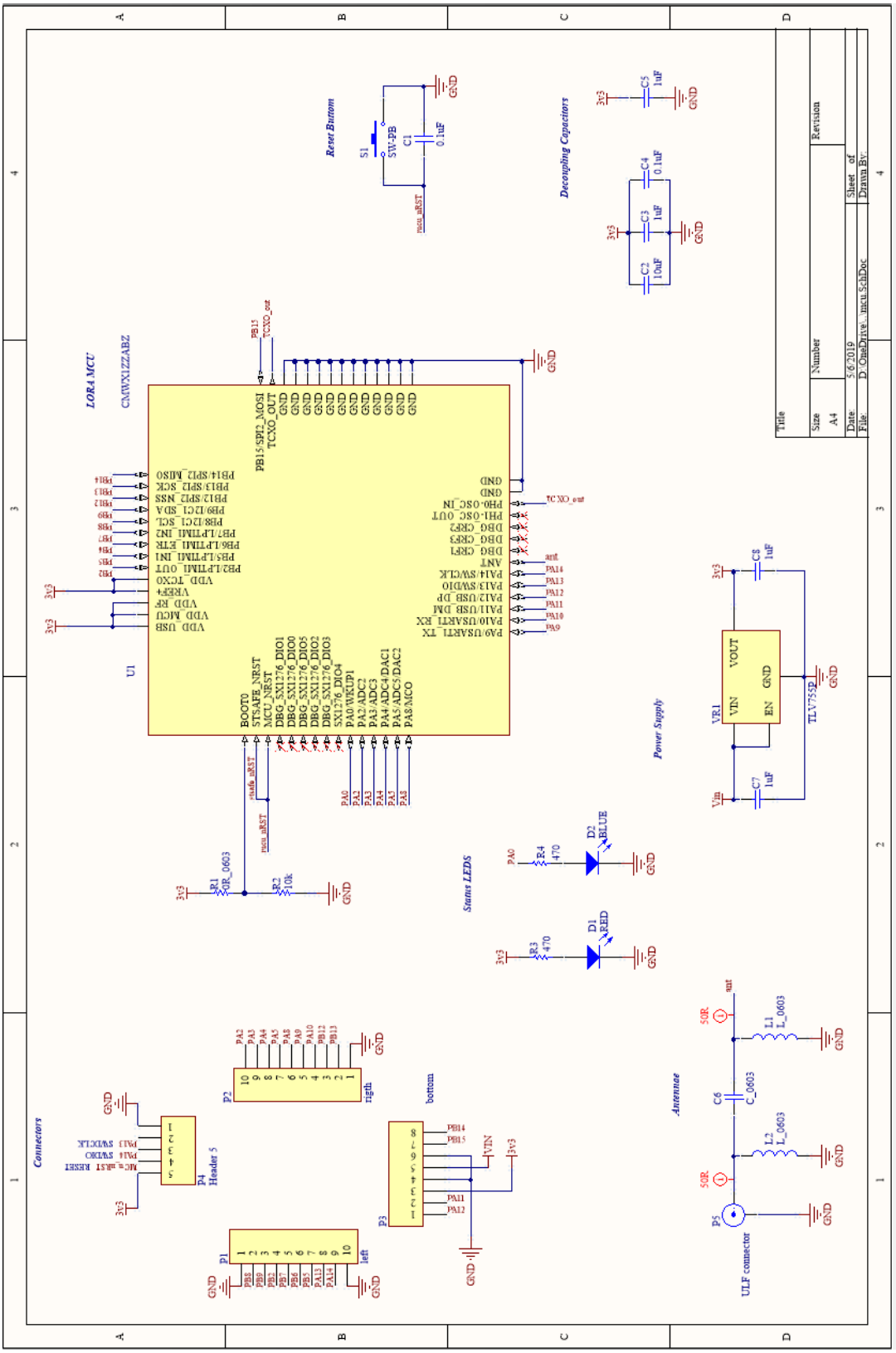


Figura C.1: Esquemáticos de la tarjeta de los nodos.

Anexo D

Esquemáticos de la placa de expansión

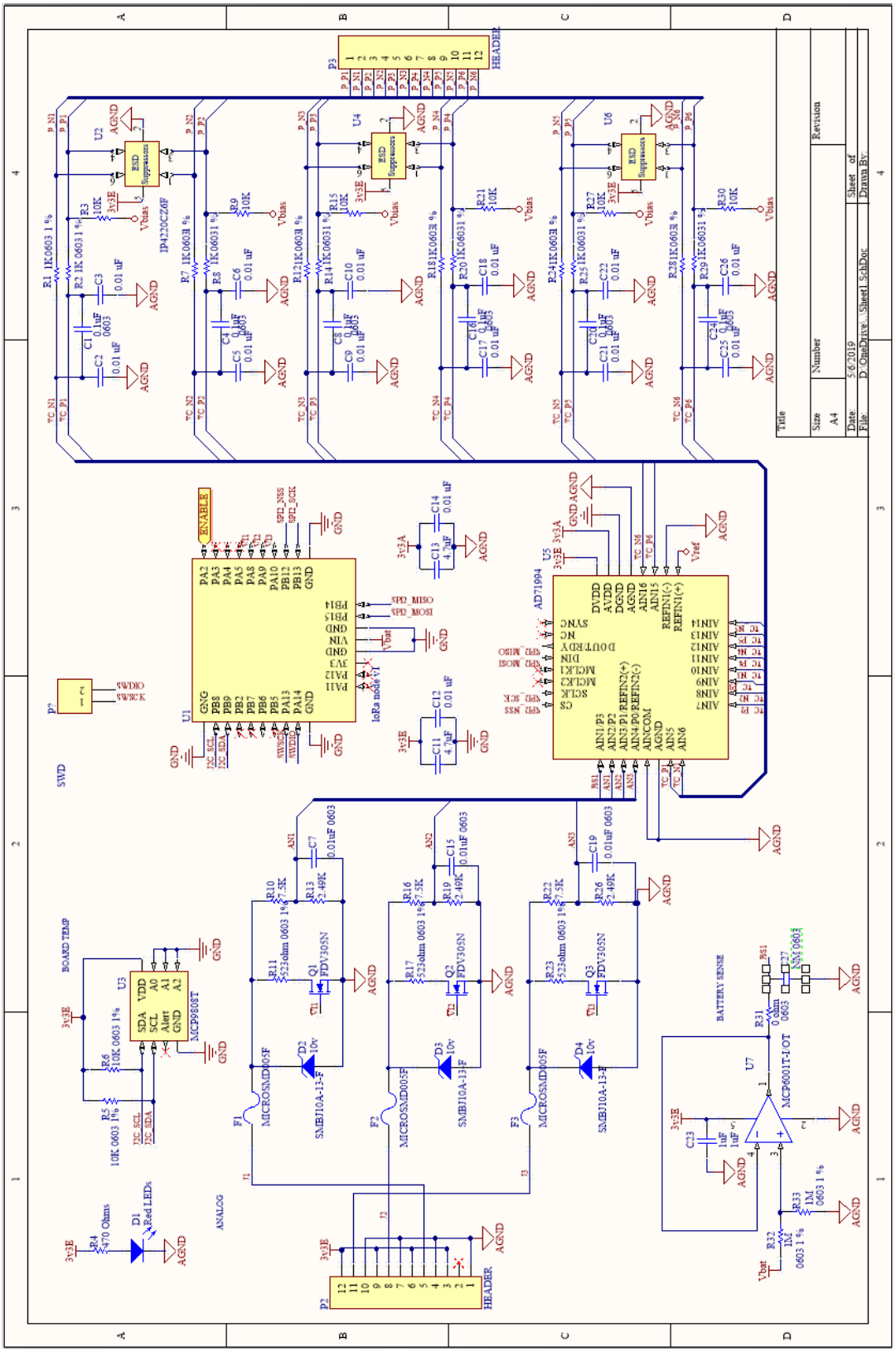


Figura D.1: Esquemáticos de la tarjeta de expansión.

Anexo E

Lista de componentes

A continuación se listan los componentes utilizados en cada una de las tarjetas.

Description	Designator	Part Number	Value
Capacitor	C1, C4	C0603C104J3RAC	0.1uF
Capacitor	C2	GRM188R60J106KE47D	10uF
Capacitor	C3, C5, C7, C8	885012206052	1uF
INFRARED GaAs LED	D1	150060RS75000	Red
INFRARED GaAs LED	D2	150060BS75000	NM
BNC Elbow Connector	P5	CONUFL001-SMD	
Resistor	R2	RC0603FR-0710KL	10K
Resistor	R3, R4	AC0603FR-13470RL	470ohm
Switch	S1	TL6330AF200Q	
RF MODULE	U1	CMWX1ZZABZ-078	
LDO Voltage Regulators	VR1	TLV75533PDBVR	

Tabla E.1: Componentes del nodo básico.

En la Tabla **??** se encuentran los componentes utilizados en la placas de los nodos y en la Tabla **??** los componentes de la placa de expansión.

Description	Designator	Part Number	Value
Capacitors	C1, C4, C8, C16, C20, C24	C0603C104J3RAC	0.1uF
Capacitors	C2, C3, C5, C6, C9, C10, C12, C14, C17, C18, C21, C22, C25, C26, C30, C31, C33	885012206089	0.01 uF
Capacitors	C7, C15, C19	885012206089	0.01uF
Capacitors	C11, C13, C34	JMK107BJ475KA-T	4.7uF
Capacitors	C23, C28, C29, C32, C35, C36	885012206052	1uF
INFRARED GaAs LED	D1	150060RS75000	10V
ESD Suppressors	D2, D3, D4	SMBJ10A-13-F	
Resettable Fuses	F1, F2, F3	MICROSMD005F-2	
Ferrite Beads	L1	2506031217Z0	
MOSFET	Q1, Q2, Q3	FDV305N	
Resistors	R1, R2, R7, R8, R12, R14, R18, R20, R24, R25, R28, R29, R36, R37	CRCW06031K00FKEAC	1K

Tabla E.2: Componentes de la placa de expansión.

Anexo F

Manual de usuario

F.1. Introducción

El sistema *LoRa Therm v1.0* es un sistema de monitorización de temperatura que utiliza un transmisor LoRa para comunicar los datos. Esté sistema utiliza un protocolo de transmisión inalámbrico propio, especialmente diseñado para su utilización en redes de sensores.

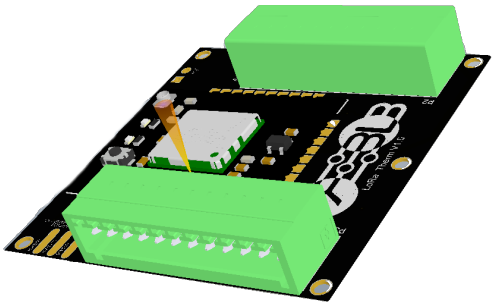


Figura F.1: LoRa Therm V1.

Para que el sistema pueda pueda transmitir los datos, debe estar conectado a una red. Una red está conformada por uno o más nodos esclavos y por un nodo central (al que llamaremos máster a partir de ahora). El máster actúa como punto de acceso y proporciona los parámetros de configuración a los nodos. El máster funciona mediante un servicio (diseñado en python) que recibe los datos del transmisor (mediante puerto serie), los procesa, los almacena y, de ser necesario, genera una respuesta.

F.2. Hardware

El sistema está conformado por dos bloques:

- **Módulo:** Es un transmisor sin aplicación definida. Está formado por un microcontrolador ARM Cortex M0+ y un transmisor de LoRa SX1276. Este hardware contiene lo mínimo indispensable para que el transmisor pueda trabajar de manera autónoma.
- **Therm Carrier:** Es una placa de expansión que implementa un módulo.

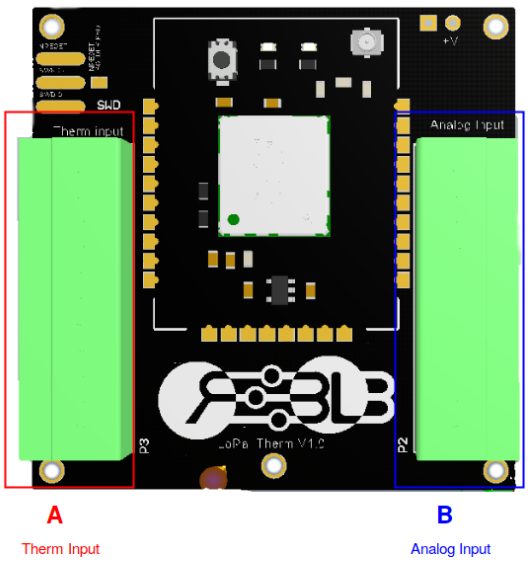


Figura F.2: Esquema del sistema.

Cuenta con 6 entradas adaptadas para termopares del tipo T, J y K. También cuenta con 3 entradas analógicas con un margen de entrada en tensión de 0 V a 10 V y de corriente de 0 mA a 20 mA.

El microcontrolador usa dos de sus interfaces SPI para comunicarse con el transceptor de radio y con el ADC. De este también se utiliza una interfaz I^2C para la comunicación con el sensor de temperatura de superficie y 4 GPIOs para la activación de la placa de expansión y para la conmutación de las entradas analógicas (lectura de tensión o corriente). Los pines del microcontrolador usados se pueden ver resumidos en la Tabla **??**.

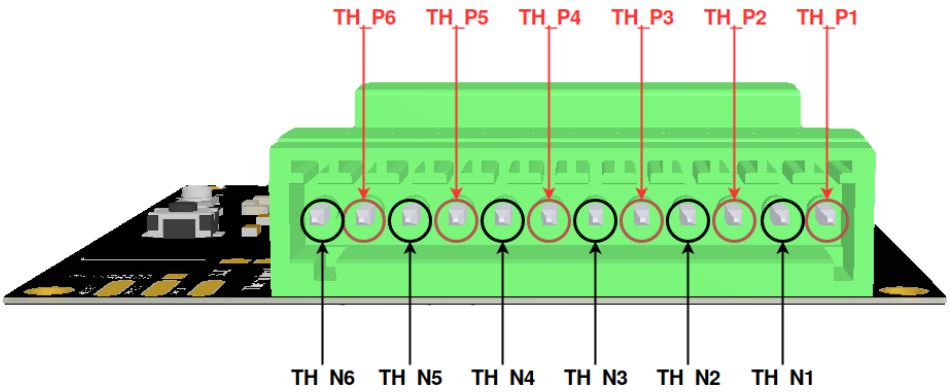


Figura F.3: Vista de los conectores de los termopares.

La tarjeta cuenta con dos conectores de 12 pines conectados a los canales de entrada del ADC. En conector A de la Figura **??** tiene las entradas las de los termopares según la asignación de la Figura **??**. El conector B de la figura **??** tiene las 3 entradas analógicas ordenadas según la Figura **??**. La relación entre las entradas y los canales del ADC se pueden ver resumidos en la Tabla **??**.

Nombre	Puerto	Pin	Descripción
MISO	B	14	Master Input/ Slave output AD7194
MOSI	B	15	Master output/ Slave Input AD7194
SCK	B	13	Master clock AD7194
CS	B	12	Chip select AD7194
SDA	B	9	I^2C Data MCP9808
SCL	B	8	I^2C Clock MCP9808
UART_TX	A	3	UART TX
UART_RX	A	3	UART RX
$\bar{V_1}$	A	5	Analog current read enable 1
$\bar{V_2}$	A	8	Analog current read enable 2
$\bar{V_3}$	A	9	Analog current read enable 3
ENABLE	A	2	Board enable
Status_led	A	0	Status led

Tabla F.1: Conexiones de los dispositivos.

Nombre	Canal	Función
TH_P1	CHANNEL_5	Entradas de termopares
TH_N1	CHANNEL_6	
TH_P1	CHANNEL_7	
TH_N1	CHANNEL_8	
TH_P1	CHANNEL_9	
TH_N1	CHANNEL_10	
TH_P1	CHANNEL_11	
TH_N1	CHANNEL_12	
TH_P1	CHANNEL_13	
TH_N1	CHANNEL_14	
TH_P1	CHANNEL_15	
TH_N1	CHANNEL_16	
AIN1	CHANNEL_2	Entradas analógicas
AN2	CHANNEL_3	
AIN3	CHANNEL_4	
BAT	CHANNEL_1	Sensor de batería

Tabla F.2: Relación entrada/canal del ADC.

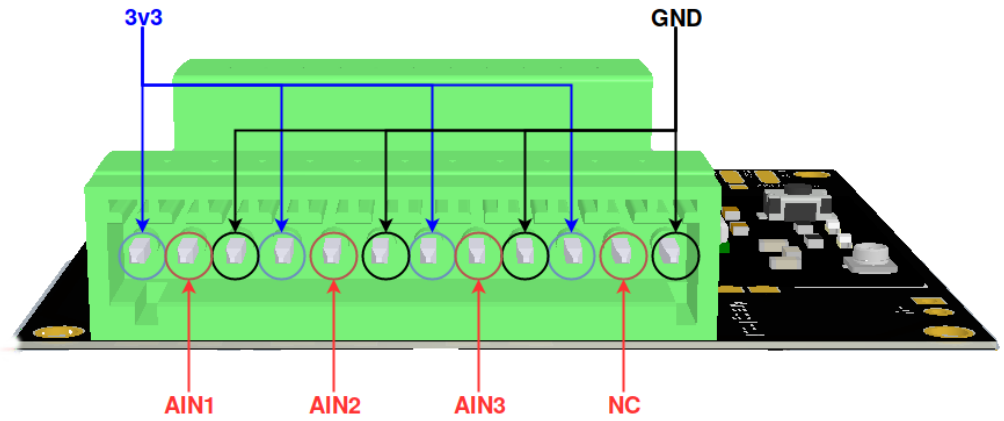


Figura F.4: Vista de los conectores de las entradas analógicas.

F.3. Base de datos y plataforma web

Los datos procedentes de los nodos se almacenas en una base de datos tipo SQL con la estructura de la figura **??**. Esta base de datos, además de los valores de los sensores, contiene información referente a los nodos, la configuración de sus sensores y los parámetros de configuración que utilizan.

Con el fin de facilitar el acceso y compresión de los datos, se ha diseñado una plataforma web. Desde esta plataforma se pueden consultar los datos de los diferentes sensores, modificar los parámetros de configuración de los nodos y modificar los parámetros de la red. Esta plataforma está diseñada en php y funciona mediante un servidor apache.

F.4. Descarga de los ficheros fuente

Los archivos necesarios pueden descargarse desde el repositorio de github. Para ello, abra una consola y ejecute el comando:

```
$ git clone https://github.com/Alejo2313/LoRa_TFG.git
```

De no tener *git* instalado puede instalarlo ejecutando:

```
$ sudo apt install git
```

Al clonar el repositorio, se descargaran todos los ficheros necesarios para poder replicar el proyecto. Dentro de la carpeta *Software* se encuentran el código fuente de los nodos y del máster; en *Hardware* se encuentran los esquemáticos y ficheros de fabricación de las tarjetas; en *web* se encuentran el servidor, la base de datos y los ficheros de la web. Por úlitmo, en la carpeta *doc* está toda la documentación referente al proyecto.

F.5. Compilación y carga de binarios

F.5.1. Instalación de dependencias

Para generar el *makefile* del proyecto es necesaria la herramienta *CMake*. Puede instalarla ejecutando:

```
$ sudo apt install cmake
```

El proyecto se compila con el conjunto de herramientas arm-gcc-none-eabi. Para instalarlas, descargue la ultima versión del compilador desde `developer.arm.com`. Una vez descargado, lo primero, será eliminar cualquier versión previa del compilador presente en el ordenador.

```
$ sudo apt remove binutils-arm-none-eabi \
    gcc-arm-none-eabi libnewlib-arm-none-eabi
```

Seguidamente abra una consola en la carpeta donde se encuentra el fichero descargado y ejecute los siguientes comandos para extaer y mover los ficheros necesarios.

```
#extraemos los binarios
$ tar -xjvf gcc-arm-none-eabi-*-linux.tar.bz2
#movemos la carpeta
$ sudo mv gcc-arm-none-eabi-*/ \
    /opt/gcc-arm-none-eabi

#Exportamos la ruta
$ echo "PATH=$PATH:/opt/gcc-arm-none-eabi/bin" >> ~/.bashrc
```

Una vez instalado el compilador, lo último será instalar la herramienta *st-util*. El código fuente y las instrucciones de instalación se pueden encontrar en el link `https://github.com/texane/stlink.git`. Esta será usada para cargar los binaros en las placas.

F.5.2. Compilación del código

Para compilar el código, primero acceda a la carpeta que contiene el código deseado. En el caso del código de los nodos:

```
$ cd software/Project/Node/
```

Seguidamente, elimine los ficheros de compilación y los binarios existentes en la carpeta *build*.

```
$ rm -rf build/
$ mkdir build
$ cd build
```

Ahora, generaremos los ficheros de complicación con cmake y compilaremos el código. para ello ejecute:

```
$ cmake ..
$ make
```

Al finalizar el proceso, verá que se han generado los ficheros *LoRaNODE.bin*, *LoRaNODE.map* y *LoRaNODE.elf*.

F.5.3. Carga de los binarios

Para cargar los binarios en la tarjeta conecte el programador *stlink* al ordenador. sitúese en la carpeta donde tiene instalada la herramienta *st-util*, abra una consola y ejecute el comando:

```
$ st-flash write [PATH TO PROJECT]/build/LoRaNODE.bin 0x8000000
```

F.6. Inicialización de la plataforma

F.6.1. Instalación de dependencias

La plataforma está basada en php, mysql y apache. Para instalarlos ejecute:

```
#install apache server
$ sudo apt-get update -y
$ sudo apt-get install apache2

#install mysql
$ sudo apt-get install mysql-server -y
$ sudo /usr/bin/mysql_secure_installation

#install php
$ sudo apt-get install php -y
$ sudo apt-get install -y  \
    php-{bcmath,bz2,intl,gd,mbstring,mcrypt,mysql,zip} \
    && sudo apt-get install libapache2-mod-php -y

$ systemctl restart apache2.service
```

F.6.2. Puesta en funcionamiento

Deberá cargar el *backup* con la estructura de la base de datos desde el fichero *node.sql*. Seguidamente deberá copiar el contenido de la carpeta *[PROJECT PATH]/web/html* en la carpeta de datos del servidor (generalmente /var/www/html/).

Para modificar las credenciales que utiliza la plataforma, abra el fichero *connect.php* en la ruta de datos de la web. Seguidamente, modifique las variables:

```
$serverName = "localhost";
$userName   = "[YOUR_SQL_USER]";
$password   = "[YOUR_SQL_PASSWORD]";
$dbnames    = "nodes";
```

F.7. Puesta en marcha del servidor

F.7.1. Instalación de dependencias

El servidor funciona con python3, para instalo, abra una consola y ejecute:

```
$ sudo apt install python3 python3-pip
```

El sistema utiliza las liberías *pyserial* y *mysql-connector*. para instalarlas, ejecute:

```
$ pip3 install pyserial mysql-connector
```

F.7.2. Puesta en funcionamiento

Primero, si no lo ha hecho antes, deberá cargar el *backup* con la estructura de la base de datos desde el fichero *node.sql*. Para poner en marcha el servidor, abra una consola en la ruta *[PROJECT PATH]/web/database/db* y ejecute el comando:

```
$ python3 tsensor.py
```

Para modificar las credenciales de acceso a la base de datos y el puerto al que está conectado el máster, deberá editar los parámetros del fichero *connect.py*

```
port = '/dev/ttyACM0'
baudrate = 115200

userName   = "[YOUR_SQL_USER]"
password   = "[YOUR_SQL_PASSWORD]"
dbnames    = "nodes"
serverName = "localhost"
```

F.8. Modificación del proyecto

F.8.1. Modificación de los parámetros por defecto

Los parámetros de configuración del proyecto se encuentan en el fichero *[CODE PATH]/Inc/system_config.h*. Los parámetros modificables y sus posibles valores se encuentran resumidos en ese fichero.

F.8.2. Modificación de la configuración del hardware

Los parámetros de configuración del hardware se encuentra en la ruta *global/Core/BOARDS/B-L072Z-LRWAN1/inc/mlm32l0xx_hw_conf.h*. Se recomienda crear una copia de seguridad antes de modificar el fichero.

F.8.3. Modificación y adición de rutas y opciones de compilación

Los ficheros de compilación son generados automáticamente por cmake. Puede añadir nuevas rutas y parámetros de configuración modificando los ficheros *STM32ToolChain.cmake* y *CMakeLists.txt*. Para obtener más información sobre cmake y sobre como añadir nuevas ruta visite la página `cmake.org`.