GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

Transmisión de pulsos de auto generadores TENG mediante protocolo LoRaWAN

Autor: Ignacio Astarloa Olaizola

Director: José Sánchez del Río Sáez

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

"Transmisión de pulsos de auto generadores TENG mediante protocolo LoRaWAN" en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2020/21 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Ignacio Astarloa Olaizola Fecha: 10 / 07 / 2021

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: José Sánchez del Río Sáez Fecha: 10 / 07 / 2021

Jose Sanchez del Rio

Firmado digitalmente por Jose Sanchez del Rio
Fecha: 2021.07.10 19:56:58 +02'00'

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

Transmisión de pulsos de auto generadores TENG mediante protocolo LoRaWAN

Autor: Ignacio Astarloa Olaizola

Director: José Sánchez del Río Sáez

Madrid

TRANSMISIÓN DE PULSOS DE AUTO GENERADORES TENG MEDIANTE PROTOCOLO LORAWAN

Autor: Astarloa Olaizola, Ignacio.
Director: Sánchez del Río Sáez, José.
Entidad Colaboradora: IMDEA Materiales

RESUMEN DEL PROYECTO

En este proyecto se ha desarrollado un sistema de comunicación basado en protocolo LoRaWAN y el servidor red de la iniciativa de The Things Network. A partir de este sistema de comunicación se han desarrollado dos funcionalidades para transmitir información de un pulso generado por un nano generador triboeléctrico o TENG (*Triboelectric Nanogenerator*)

Palabras clave: Internet de las cosas, LoRaWAN, TENG, transmisión.

1. Introducción

La transición hacia un mundo conectado en el que se pueda hacer la realidad el Internet de las Cosas o IoT, pasa por conseguir métodos de comunicación de bajo coste, bajo consumo y alta fiabilidad. LoRaWAN es un protocolo que surge para satisfacer esta necesidad y que se caracteriza por esas tres cualidades además de permitir comunicaciones en muy largas distancias, pudiendo llegar a superar los 500 km en casos ideales (Allan, 2019). Como se trata de un protocolo de comunicación orientado al IoT, una vez la información es transmitida desde el dispositivo emisor, esta información es accesible desde cualquier parte del mundo (siempre que se tenga conexión a internet). De manera paralela es importante encontrar dispositivos que no requieran de mucho consumo de energía o que puedan autogenerarla y que sirvan como sensores capaces de detectar información. Por este motivo han surgido los TENG que son nano generadores fabricados con materiales altamente capacitivos y capaces de generar pulsos eléctricos a partir de estímulos químicos o mecánicos.

2. Definición del proyecto

En este proyecto se ha diseñado un sistema de comunicación usando tecnología LoRa y basado en el protocolo LoRaWAN. Para la transmisión de la información es necesario un dispositivo final que emita la información, una pasarela o gateway que reciba esta información y convierta los paquetes de radiofrecuencia en paquetes IP, un servidor de red que organice la comunicación de los dispositivos finales con los gateways y los gateways con internet y un servidor de aplicación que permita almacenar la información transmitida y acceder a ella.

Para las funcionalidades, ha sido necesario el diseño de un programa que cumpliese con los requerimientos buscados, la prueba empírica de estos programas, y el análisis de los resultados obtenidos. La primera funcionalidad simplemente buscaba transmitir un mensaje de alarma cuando se alcance un umbral de tensión en el pulso generado por

el TENG. La segunda funcionalidad busca la transmisión íntegra de uno o varios pulsos de manera que se pueda analizar la forma, tensión de pico y frecuencia una vez se recibe la información.

3. Descripción del sistema de comunicación

Para el servidor de aplicación se ha empleado la iniciativa de The Things Network que consiste en una comunidad de usuarios donde existe un servidor de red ya creado. Con The Things Network no es necesaria la creación de una red LoRaWAN propia y todos los procesos difíciles que esto conlleva (manejo de paquetes duplicados desde múltiples puertas de enlace, derivación de datos a servidores, manejo de uniones, etc.). Además, con The Things Network se garantiza una red altamente segura que soporta una verdadera encriptación de extremo a extremo, mitigaciones contra varios ataques de hackers en el camino y soporte para diferentes claves de encriptación de 128 bits para cada dispositivo final.

Para el diseño del sistema de comunicación se utilizado únicamente un tipo de dispositivo final, el Arduino MKR WAN 1300. Para poder utilizarlo se ha programado y dado de alta en The Things Network. Es en este dispositivo donde se han diseñado los dos programas para las distintas funcionalidades. También se han programado dos decodificadores distintos para poder entender el envío en su recepción ya que el mensaje se transmite en lenguaje hexadecimal. En la Ilustración 1 se refleja cómo se alimenta la placa, como se conecta el TENG a esta y cómo se utiliza una antena dipolo para aumentar el rango de transmisión.

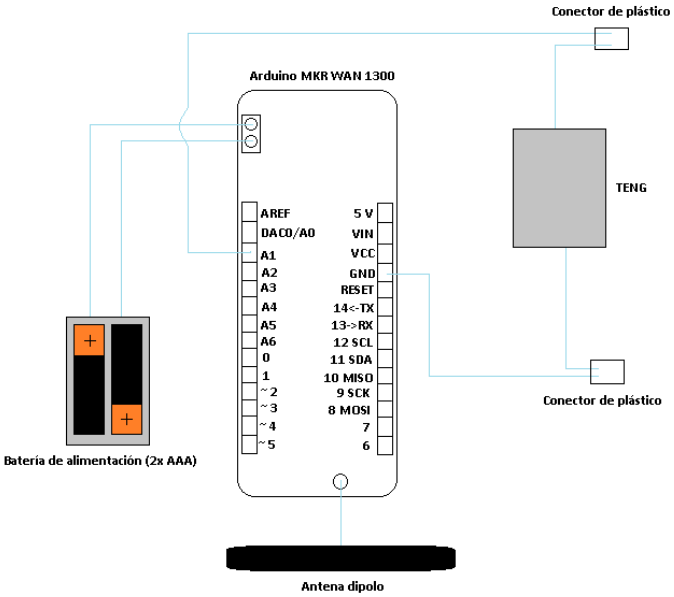


Ilustración 1. Esquemático del montaje del dispositivo final Arduino MKR WAN 1300

Como puerta de enlace o gateway se han desarrollado dos versiones, uno conectado a internet por wifi y otro por cable ethernet. Para el gateway con conexión por cable

ethernet se ha utilizado el módulo radio concentrador RAK 831 y el ordenador que actúa como host Raspberry Pi 2 Model B. Se refleja en la Ilustración 2 el montaje del primer gateway que se ha utilizado.

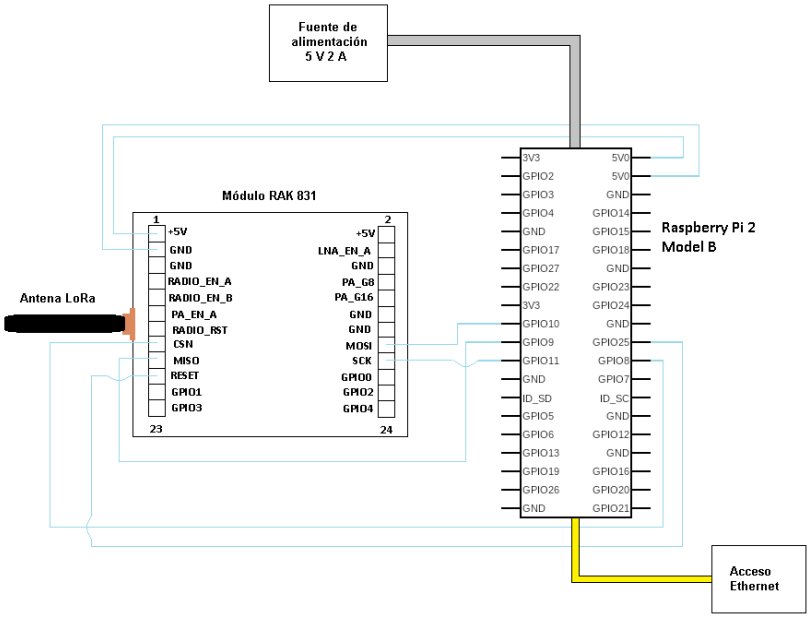


Ilustración 2. Esquemático del montaje del gateway con RAK 831 y Raspberry Pi 2 Model B

Para el segundo gateway, de conexión wifi, se ha utilizado el módulo radio concentrador RAK 2245 y el ordenador que actúa como host Raspberry Pi 4 Model B. Se refleja en la Ilustración 3 el montaje de este segundo gateway.

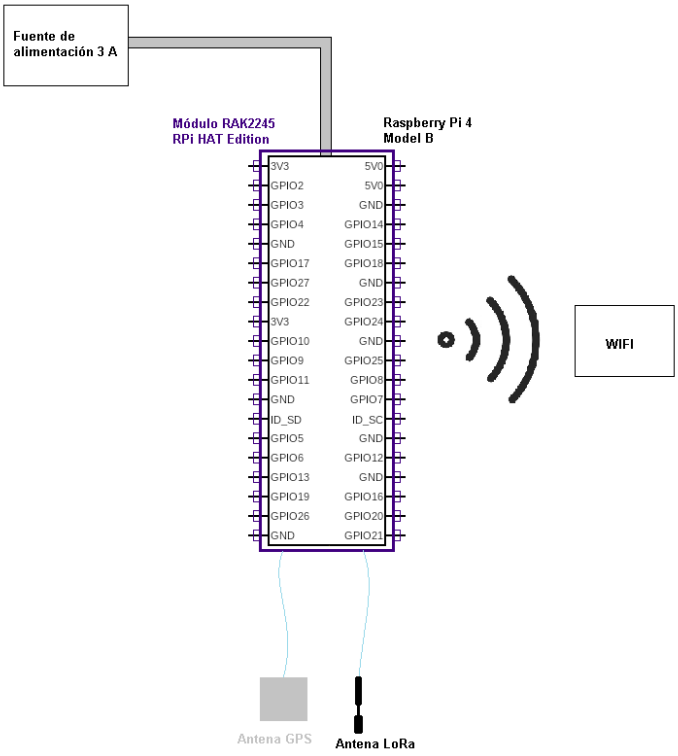


Ilustración 3. Esquemático del montaje del gateway con RAK 2245 y Raspberry Pi 4 Model B

4. Resultados

Una vez se consiguió poner en funcionamiento el sistema de comunicación se desarrollaron dos programas que permitiesen utilizar el TENG como sensor.

En el primero de ellos, más simple, se programó el Arduino MKR WAN 1300 para que cuando detectase por la entrada a la que estaba conectado el TENG una tensión superior a un umbral modificable, se enviase un mensaje de alarma. En la Ilustración 4 se puede observar cómo se recibe este mensaje en la plataforma de The Things Network accesible desde cualquier parte del mundo.

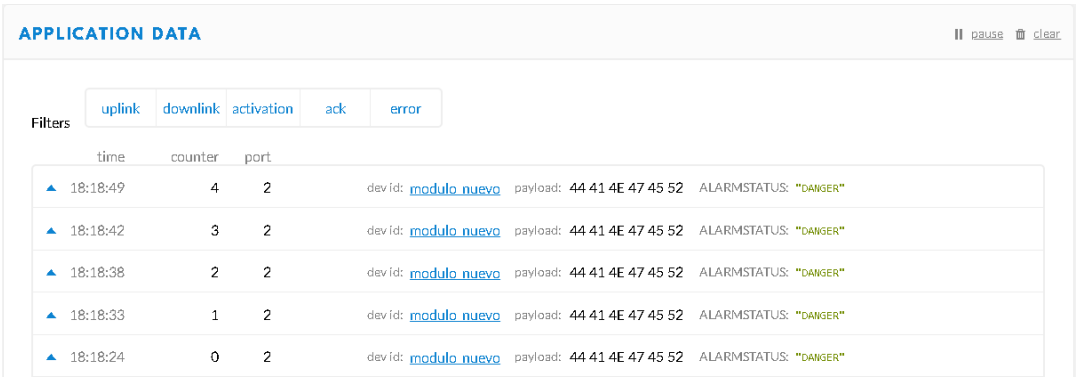


Ilustración 4. Ejemplo de transmisión de mensaje de alarma

Para la segunda funcionalidad se trabajó en un programa que digitalizase el pulso por una de las entradas ADC del Arduino, grabase los datos y los enviase en paquetes de pulsos muestreados para luego poder ser representados en Excel y comparados con los originales. En la Ilustración 5 se muestra un ejemplo de un pulso transmitido super puesto con el pulso original obtenido a partir del osciloscopio.

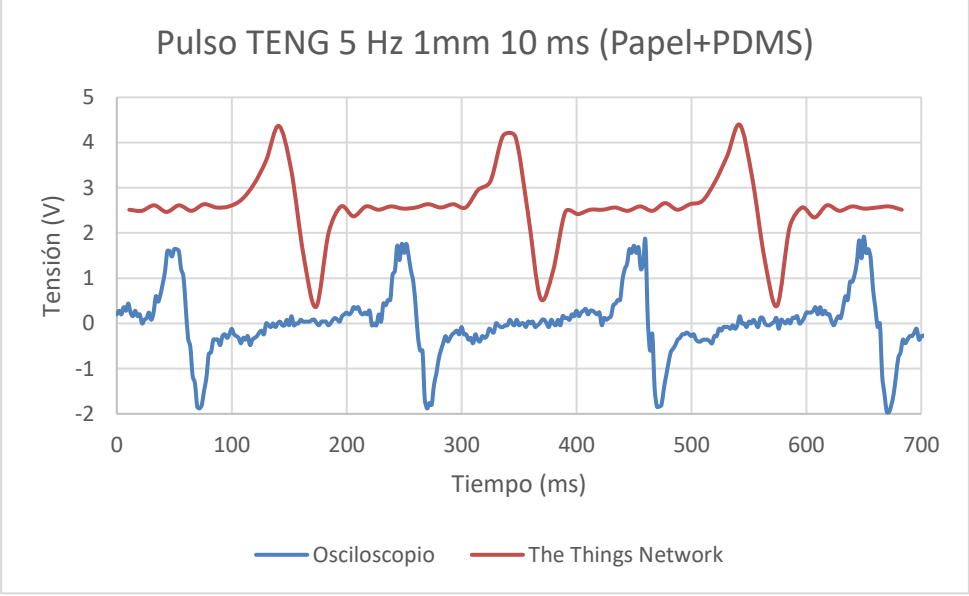


Ilustración 5. Ejemplo de pulsos generados por TENG a partir de osciloscopio y de transmisión LoRaWAN

5. Conclusiones

El protocolo de comunicaciones tiene muchísimo potencial por su fiabilidad, seguridad, distancia y poco consumo. Para transmisiones de poca frecuencia de envío de mensajes y mensajes con poco tamaño es un sistema ideal. Por ejemplo, en la funcionalidad del mensaje de alarma no ha habido ninguna dificultad. Cuando se busca aumentar el tamaño de los mensajes o realizar envíos con mayor frecuencia surgen algunas limitaciones debidas a la Política de Uso Justo de LoRaWAN. Otro aspecto mejorable es la distorsión que puede añadir el MKR WAN 1300 en la conversión analógica-digital, añadiendo tensiones de ruido constantes y saturaciones. Sin embargo, en el envío no se produce ninguna distorsión y lo que se envía se recibe tal cual por tratarse de comunicación digital

6. Referencias

[1] Allan, A. (2019). *A New LoRaWAN Distance Record*. Obtenido de Medium: https://aallan.medium.com/a-new-lorawan-distance-record-577c8bc11d7b (Fecha de último acceso: 11/07/2021)

TRANSMISSION OF PULSES FROM TENG NANO GENERATORS VIA LORAWAN PROTOCOL

Author: Astarloa Olaizola, Ignacio.
Supervisor: Sánchez del Río Sáez, José.
Collaborating Entity: IMDEA Materiales

ABSTRACT

In this project, a communication system based on LoRaWAN protocol, and the network server of The Things Network initiative has been developed. From this communication system, two functionalities have been developed to transmit information from a pulse generated by a triboelectric nano generator (TENG).

Keywords: Internet of Things, LoRaWAN, TENG, transmission

1. Introduction

The transition to a connected world in which the Internet of Things, or IoT, can become a reality requires low-cost, low-power and highly reliable communication methods. LoRaWAN is a protocol that has emerged to meet this need and is characterised by these three qualities as well as enabling communications over very long distances, ideally exceeding 500 km (Allan, 2019). As it is an IoT-oriented communication protocol, once the information is transmitted from the transmitting device, this information is accessible from anywhere in the world (as long as there is an internet connection). At the same time, it is important to find devices that do not require a lot of energy consumption or that can self-generate energy and serve as sensors capable of detecting information. For this reason, TENGs have emerged, which are nano-generators made of highly capacitive materials capable of generating electrical pulses from chemical or mechanical stimuli.

2. Description of the project

In this project, a communication system has been designed using LoRa technology and based on the LoRaWAN protocol. For the transmission of information, it is necessary to have an end device that emits the information, a gateway that receives this information and converts the radio frequency packets into IP packets, a network server that organises the communication of the end devices with the gateways and the gateways with the Internet, and an application server that allows the information transmitted to be stored and accessed.

For the functionalities, it has been necessary to design a programme that fulfils the requirements sought, the empirical testing of these programmes, and the analysis of the results obtained. The first functionality simply sought to transmit an alarm message when a voltage threshold is reached in the pulse generated by the TENG. The second functionality seeks to transmit one or several pulses in their entirety so that the shape, peak voltage and frequency can be analysed once the information is received.

3. Description of the communication system

For the application server, The Things Network initiative has been used, which consists of a community of users where a network server has already been created. With The Things Network there is no need to create your own LoRaWAN network and all the difficult processes that this entails (handling duplicate packets from multiple gateways, forwarding data to servers, handling junctions, etc.). In addition, with The Things Network a highly secure network is guaranteed that supports true end-to-end encryption, mitigations against various hacker attacks along the way and support for different 128-bit encryption keys for each end device.

For the design of the communication system only one type of end device was used, the Arduino MKR WAN 1300. In order to use it, it has been programmed and registered in The Things Network. It is on this device that the two programs for the different functionalities have been designed. Two different decoders have also been programmed to be able to understand the sending and receiving of the message, as the message is transmitted in hexadecimal language. Illustration 1 shows how the board is powered, how the TENG is connected to it and how a dipole antenna is used to increase the transmission range.

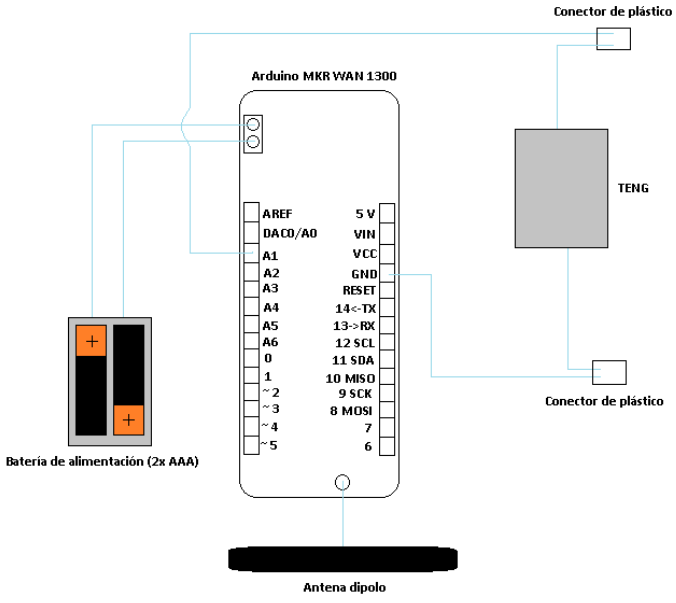


Illustration 1. Schematic of the assembly of the Arduino MKR WAN 1300 final device

Two versions have been developed as a gateway, one connected to the internet via wifi and the other via Ethernet cable. For the gateway with an Ethernet cable connection, the RAK 831 radio concentrator module and the Raspberry Pi 2 Model B host computer have been used. Illustration 2 shows the assembly of the first gateway used.

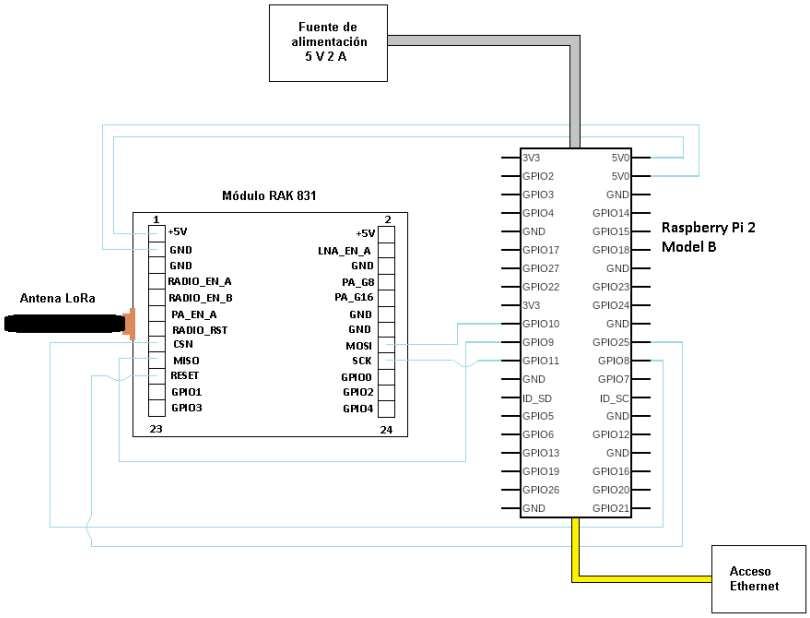


Illustration 2. Schematic of the gateway assembly with RAK 2245 and Raspberry Pi 4 Model B

For the second gateway, with wifi connection, the RAK 2245 radio concentrator module and the Raspberry Pi 4 Model B host computer have been used. The assembly of this second gateway is shown in Illustration 3.

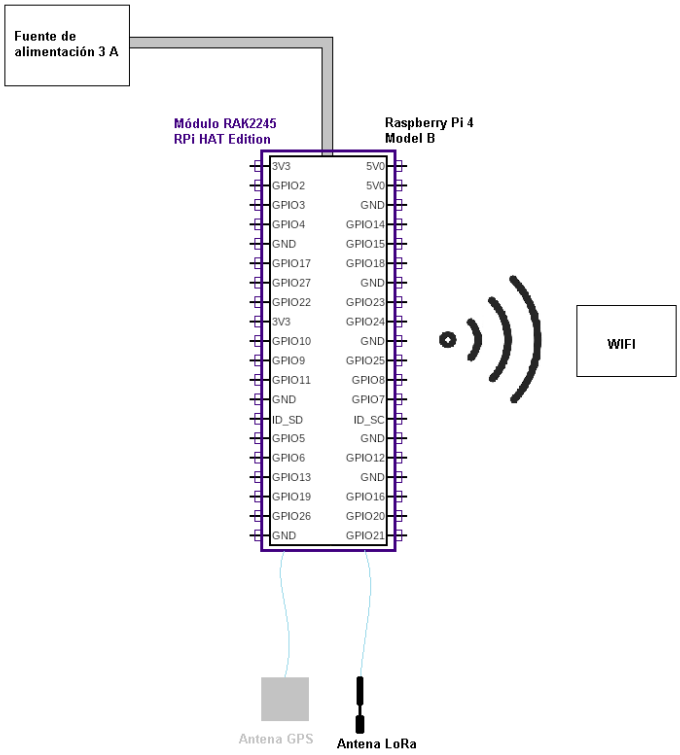


Illustration 3. Schematic of the gateway assembly with RAK 2245 and Raspberry Pi 4 Model B

4. Results

Once the communication system was up and running, two programs were developed to use the TENG as a sensor.

In the first of these, which is simpler, the Arduino MKR WAN 1300 was programmed so that when it detected a voltage above a modifiable threshold at the input to which the TENG was connected, an alarm message would be sent. Illustration 4 shows how this message is received on The Things Network platform, accessible from anywhere in the world.

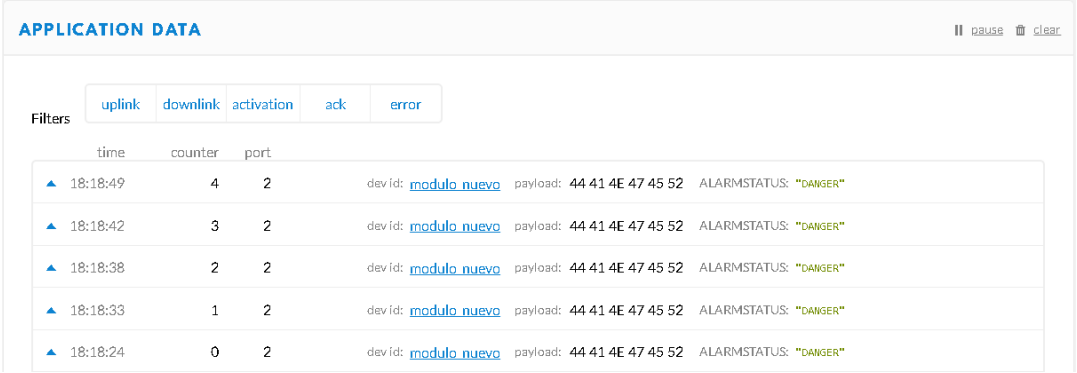


Illustration 4. Example of alarm message transmisión

For the second functionality, we worked on a program that digitised the pulse through one of the ADC inputs of the Arduino, recorded the data and sent it in packets of sampled pulses so that they could be represented in Excel and compared with the original ones. Illustration 5 shows an example of a transmitted pulse superimposed with the original pulse obtained from the oscilloscope.

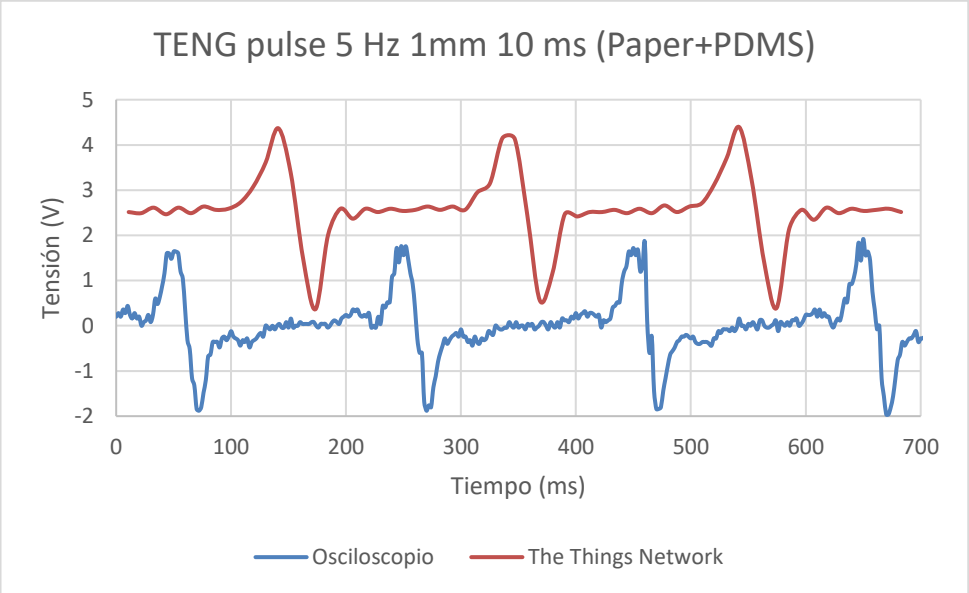


Illustration 5. Example of TENG-generated pulses from oscilloscope and LoRaWAN transmission

5. Findings

The communications protocol has great potential due to its reliability, security, distance, and low power consumption. For infrequent message transmissions and small message sizes, it is an ideal system. For example, there have been no difficulties with alarm message functionality. When looking to increase message size or send more frequent messages, some limitations arise due to LoRaWAN's Fair Use Policy. Another area for improvement is the distortion that the MKR WAN 1300 can add in the analogue-to-digital conversion, adding constant noise voltages and saturations. However, there is no distortion on the send side and what is sent is received as is because it is digital communication.

6. References

[1] Allan, A. (2019). *A New LoRaWAN Distance Record*. Obtenido de Medium: https://aallan.medium.com/a-new-lorawan-distance-record-577c8bc11d7b (Fecha de último acceso: 11/07/2021)

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. INTRODUCCIÓN

Este proyecto surge de la motivación de avanzar cada día más hacia un mundo conectado gracias al "Internet de las Cosas". Tanto el protocolo de comunicación que se utiliza como el propio nano generador que proporciona el pulso a transmitir tienen aplicaciones por separado en el mundo de la conectividad electrónica.

Los protocolos de comunicación para *IoT* son aún nuevos y desconocidos y por este motivo en este trabajo se busca explicar su funcionamiento y el extenso abanico de aplicaciones que pueden derivar de ellos. Para un futuro en el que las ciudades conectadas o las carreteras inteligentes estén a la orden del día, es necesario que se siga investigando en el campo de la comunicación inalámbrica.

Por otro lado, la investigación de nuevas fuentes de energía eléctrica limpia es vital para poder alimentar todos los componentes electrónicos que hacen falta para lograr un futuro conectado y a la vez sostenible. Los generadores TENG son cada vez más populares ya que se son los dispositivos con mayor capacidad de autogeneración de potencia eléctrica. Por este motivo, existen proyectos que se han centrado en desarrollar estrategias prometedoras para lograr TENGs de alto rendimiento orientados a aplicaciones prácticas (Liu, y otros, 2019). Además, los TENGs también presentan la posibilidad de ser usados como detectores o sensores de estímulos mecánicos, cómo se mostrará en la sección Parte I5.3.1.

Por todo esto, este proyecto resulta atractivo para un estudiante de ingeniería industrial especializado en electrónica que tiene ganas de aprender y conocer más sobre el futuro de la conectividad.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

2.1 NANO GENERADOR TRIBOELÉCTRICO (TENG)

La primera tecnología que es necesaria introducir es el nano generador triboeléctrico, al que se referirá cómo TENG (*Triboelectric Nanogenerator)*. Los TENGs surgen a principios del siglo pasado y representan una nueva forma de generación de energía eléctrica a partir de estímulos químicos o mecánicos. El éxito de los TENG reside en la posibilidad que ofrecen de convertir eficazmente la energía mecánica cotidiana en energía eléctrica útil. No sólo la energía natural, como la lluvia o el viento, sino también los movimientos corporales cotidianos, como tocar con la mano o caminar, pueden servir como fuente de energía mecánica para el efecto triboeléctrico. En los dispositivos TENG, las cargas se separan en las superficies de contacto y se genera un potencial eléctrico entre ellas. El potencial resultante de los movimientos mecánicos dinámicos puede acumularse en una unidad de almacenamiento o utilizarse para alimentar dispositivos eléctricos (Kim, Lee, Kim, & Jeong, 2020). Las superficies de contacto pueden estar compuestas de diferentes materiales dependiendo las características del generador que se quieran optimizar (tensión máxima a generar, vida útil, resistencia a presiones mecánicas, etc.) pero en el desarrollo de este trabajo se ha experimentado con TENGs fabricados con papel y polidimetilsiloxano y con papel y poliamida.

Como se ilustra en la Figura 1, cuando las superficies entran en contacto se genera un pico de tensión positivo, y cuando éstas se separan se genera un pico de tensión negativo. El conjunto de ambos picos conforma el pulso que se desea transmitir mediante el protocolo de comunicación que se explica en la sección 2.2.

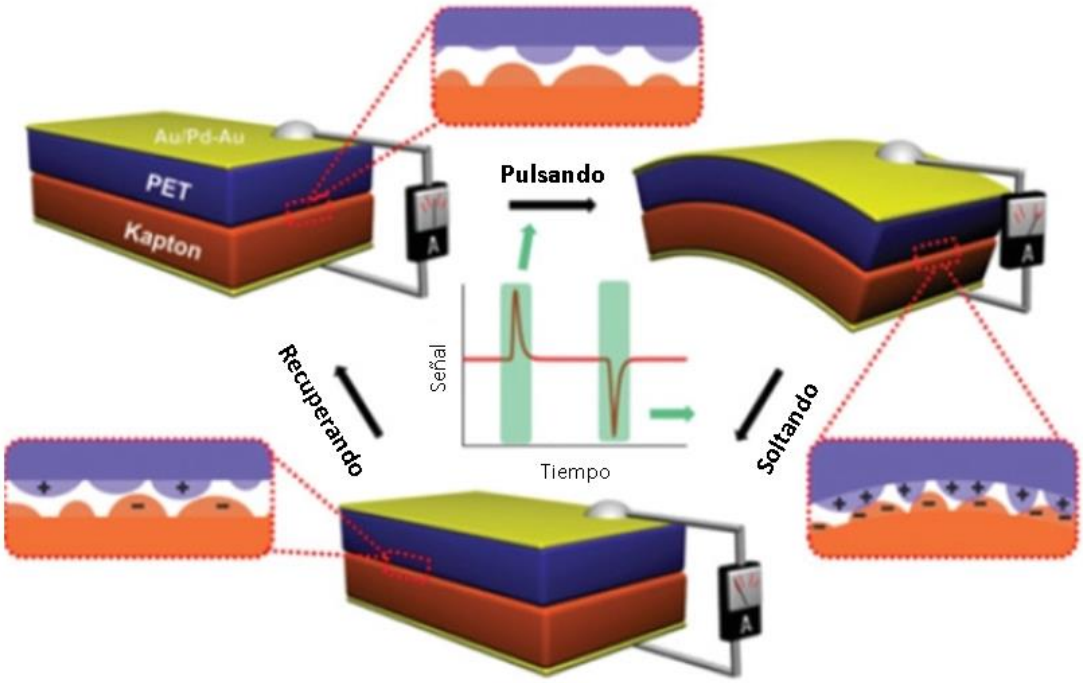


Figura 1. Explicación del funcionamiento de un TENG (Wu, Wang, Ding, Guo, & Wang, 2019)

En este trabajo no se va a estudiar de manera más detallada el funcionamiento de los TENGs ya que el objetivo de éste es la transmisión del pulso que generan.

2.2 TECNOLOGÍA Y PROTOCOLO LORAWAN

En este apartado se busca explicar que son la tecnología LoRa y la especificación de redes LoRaWAN y como juntas constituyen el sistema de comunicación que se ha empleado en este proyecto para la transmisión del pulso. Se trata de un sistema de comunicaciones inalámbricas de largo alcance baja potencia y baja tasa de bits, promovido por "*Lora Alliance*".

Por un lado, la tecnología LoRa, que proviene de "Largo Rango" en inglés *"Long Range"*, representa el primero de los niveles de OSI, la capa física. Permite envíos y recepciones de larga distancia con dispositivos de tamaño reducido y bajo consumo. En situaciones cotidianas, esta tecnología funciona sin ningún problema hasta los 15 km, sin embargo, el récord está fijado en 766 km (las condiciones para alcanzar estas distancias tienen que ser extremadamente favorables) (Allan, 2019) Para lograr estas distancias, utiliza una técnica conocida como "espectro ensanchado". Esta estrategia consiste en mandar una señal con más ancho de banda que el teórico necesario para permitir la recepción de múltiples señales al mismo tiempo, aunque éstas tengan distintas velocidades (LoRaWAN, 2021). La tecnología fue desarrollada por *Semtech* pero es la fundación *Lora Alliance* la que se está encargando de popularizarla y evolucionarla. Una de las principales ventajas de la radio frecuencia LoRa es que opera en la banda ISM (bandas de radio internacionales y que se encuentran reservadas para el uso de energía de radiofrecuencia (RF) para fines médicos, industriales y científicos distintos de las telecomunicaciones). Estas frecuencias permiten a cualquier persona utilizar esta tecnología sin ningún tipo de licencia. Las frecuencias varían dependiendo del continente, operándose en Europa (lugar donde se ha realizado el proyecto) con una frecuencia de 868 MHz (LoRaWAN, 2021).

El segundo de los niveles OSI (red) lo representa LoRaWAN, especificación de redes que pertenece a la familia de las de comunicación de baja potencia y amplia área conocidas cómo LPWAN(*Low Power Wide Area Network*). LoRaWAN es un protocolo MAC (*Media Access Control)*, construido para utilizar la capa física de LoRa. Está diseñado principalmente para redes de sensores, en los que los sensores intercambian paquetes con

el servidor con una baja tasa de datos y con intervalos de tiempo relativamente largos. En este trabajo se llevan los límites del protocolo al extremo buscando la mayor tasa de datos permitida y el menor intervalo de tiempo disponible (Augustin, Yi, Clausen, & Townsley, 2016).

Juntos conforma lo que habitualmente se denomina el protocolo LoRaWAN. Surge por la necesidad de un sistema de comunicación que cumpla los requisitos clave del "Internet de las cosas", referido normalmente como IoT(*Internet of Things*). Está diseñado para conectar de manera inalámbrica objetos, normalmente alimentados por baterías, a internet en redes regionales, nacionales o mundiales. Las ventajas que ofrece este sistema son la posibilidad de comunicación bidireccional, la seguridad de extremo a extremo, la movilidad, el bajo consumo y los servicios de localización.

Para comprender el funcionamiento de este protocolo es importante diferenciar sus cuatro elementos principales que se explican brevemente a continuación y se ilustran en la siguiente Figura 2:

- Gateways: las pasarelas o gateways están conectadas al servidor de red mediante conexiones IP estándar (ethernet o wifi) y actúan como un puente transparente, convirtiendo simplemente los paquetes de radiofrecuencia en paquetes IP y viceversa. Pueden hacer esta función para miles de nodos finales siempre que se encuentren en un radio de alrededor de 15 km en situaciones normales o en un radio mucho mayor en condiciones muy favorables como podría ser el espacio.
- Nodos finales: se tratan de dispositivos a los que se conectan sensores (el TENG en nuestro caso), que suelen estar alimentados con baterías que les proporcionan una larga vida útil. Mandan los datos a la aplicación y reciben información de esta (todo a través de los gateways). Lógicamente deben de ser dispositivos que incorporen tecnología de radio frecuencia LoRa.
- Servidor de red: establece la conexión de diferentes puertas de enlace mediante una conexión IP/TCP (puede ser inalámbrica o por cable). Organiza qué gateway es el responsable de transmitir el mensaje cuando lo envía un nodo final, elimina los mensajes duplicados y se encarga de la gestión de las velocidades de transmisión de

datos con velocidad de datos adaptable (ADR) de los nodos finales con el objetivo de alargar la vida útil de sus baterías y maximizar la capacidad de la red.

- Servidor de aplicación: este servidor es a donde van destinados todos los datos recopilados de los nodos finales de manera que sean almacenados y analizados y se determinen las acciones de los nodos finales (Haxhibeqiri, Poorter, Moerman, & Hoebeke, 2018).

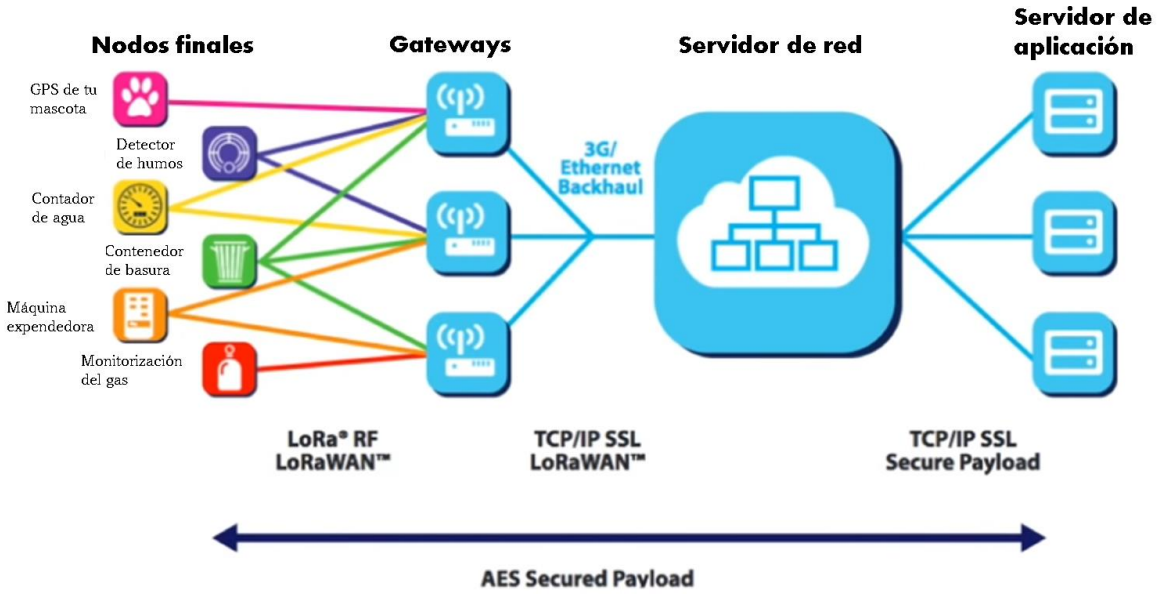


Figura 2. Explicación gráfica de los elementos del protocolo LoRaWAN (LoRaWAN, 2021)

2.3 THE THINGS NETWORK (TTN)

The Things Network es una iniciativa que nace en 2015 en Ámsterdam gracias a los holandeses Wienke Giezeman y Johan Stokking. Los fundadores buscaban la creación de una red de datos de IoT creada por las personas y que fuese gratis y abierta para que cualquiera pudiera utilizarla. Actualmente, The Things Network es una comunidad global que sigue construyendo una red LoRaWAN descentralizada y de código abierto. Es la red LoRaWAN más popular hoy en día con 146799 usuarios y 20055 gateways registrados. En la siguiente Figura 3 se muestran todos los gateways registrados públicos registrados que pueden ser usados por los distintos usuarios.

Figura 3. Mapa con la localización de todos los gateways registrados en The Things Network (TheThingsNetwork, 2021)

Puede surgir la duda de por qué utilizar The Things Network como servidor de red ya que una de las ventajas de LoRaWAN en comparación con otros protocolos como NB-IOT y Sigfox es que se puede ser el propio operador de la red; por lo tanto, no estar sujeto a las restricciones impuestas por operadores de red específicos. Así que, si se persigue crear los propios acuerdos de nivel de servicio, no tener que seguir la política de uso justo de The Things Network, etc., entonces se puede crear libremente una red propia estableciendo las propias reglas. (siempre que se sigan las reglas de tu país en cuanto a la velocidad de transmisión, etc.). Sin embargo, The Things Network proporciona un servidor de red que

se encarga de la parte complicada de la creación de una red LoRaWAN (manejo de paquetes duplicados desde múltiples puertas de enlace, derivación de datos a servidores, manejo de uniones, etc.). Además, con The Things Network se garantiza una red altamente segura que soporta una verdadera encriptación de extremo a extremo, mitigaciones contra varios ataques de hackers en el camino y soporte para diferentes claves de encriptación de 128 bits para cada dispositivo final. Otra ventaja que proporciona pertenecer a la comunidad de The Things Network es la disponibilidad de gateways de otros usuarios, aunque como ya se ha comentado, en este trabajo se van a utilizar gateways montados personalmente (TheThingsNetworkOverview, 2021).

A la hora de utilizar esta plataforma para la comunicación LoRaWAN hay que crearse una cuenta y registrar una aplicación donde registrar los nodos finales que vayan a estar asociados a esta aplicación. De igual manera si se va a trabajar con un gateway de fabricación propia es necesario también el registro de este en la página web. Estos procesos serán explicados más adelante en las secciones 5.1.2, Parte I5.2.1.3 y Parte I5.2.2.3.

Uno de los mayores problemas que presenta TTN es la Política de Uso Justo o *Fair Use Policy* que limita la velocidad de datos DR (definida por el factor de dispersión o *Spreading Factor* SF y el ancho de banda o *Bandwidth* BW) y el tamaño máximo de los paquetes. Ambos dependen aproximadamente de la distancia al gateway más cercano, y también se definen en la especificación de cada región. Al igual que para la banda europea de 863-870MHz, el tamaño máximo de los paquetes de aplicación varía:

- 51 bytes para las velocidades de datos más lentas, SF10, SF11 y SF12 en 125kHz
- 115 bytes para SF9 en 125kHz
- 222 bytes para las velocidades más rápidas, SF7 y SF8 en 125kHz (y SF7 en 250kHz)

Hay que tener en cuenta que el protocolo LoRaWAN añade al menos 13 bytes a la carga útil de la aplicación para la encriptación del mensaje (B., 2021). Por tanto, usando un SF7 en 125 kHz se debería permitir un tamaño máximo de 222-13=209 bytes. Sin embargo, algunas librerías, como la LMiC 270, sólo admiten 51 bytes para todas las velocidades de datos, algunos proveedores, como el holandés KPN 137, sólo admiten 51 bytes para todas

las velocidades de datos y algunas regiones definen un tiempo máximo de permanencia, lo que reduce el tamaño máximo de la carga útil para las velocidades de datos lentas (alta SF). Por estos motivos, es importante entender que el máximo de 209 bytes no es una verdad absoluta y existen librerías que reducen este límite de manera considerable. En la sección 5.3.2 se volverán a mencionar estos límites impuestos por las librerías cuando el valor máximo de bytes en la práctica no coincida con el máximo teórico de 209 bytes.

Capítulo 3. ESTADO DE LA CUESTIÓN

Los protocolos de comunicación LoRaWAN, al igual que el resto de las alternativas que están surgiendo actualmente para proporcionar viabilidad al concepto *Internet of Things (IoT)*, son aún desconocidos ya que son tecnologías muy jóvenes que no han tenido tiempo de ser divulgadas. Los trabajos y artículos académicos que primero comenzaron a trabajar con dispositivos LoRa y protocolos LoRaWAN no aparecen hasta mitades del siglo pasado. Primero ha sido necesaria una corriente de trabajos que se ha dedicado a estudiar la tecnología, realizar análisis de ésta, evaluar sus fortalezas y sus debilidades y explicar su funcionamiento. Es ahora cuando se están desarrollando trabajos y proyectos más centrados en las aplicaciones que pueden surgir de este sistema de comunicación. Generalmente los estudios están más orientados a transmisiones de poca información en intervalos de tiempo mucho más grandes que los que se propone en este trabajo, que es para lo que se diseñó el protocolo LoRaWAN en un inicio. En este proyecto se busca probar las capacidades del sistema de comunicación en sus condiciones límites buscando máxima cantidad de información transmitida en intervalos de tiempo más reducidos.

Los nano generadores TENG están también a la orden del día debido que suponen una revolución en la generación de energía. Ya se ha trabajado con los TENGs en proyectos de transformación de energía eólica, energía proporcionada por gotas de lluvia, energía ultrasónica y energía de las olas del agua. Ésta última se conoce como “energía azul” y ha levantado mucho interés ya que, de poder perfeccionar la cosecha de energía producida por los océanos, los TENG pueden sustituir a los generadores electromagnéticos (EMG) que no son tan eficaces en energías de vibraciones de baja frecuencia (inferiores a 5 Hz) (Wu, Wang, Ding, Guo, & Wang, 2019).

También se está investigando paralelamente la posibilidad de utilizar TENGs como sensores capaces de captar distintos estímulos mecánicos y transmitir esta información por diferentes sistemas de comunicación inalámbrica.

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

Como se ha comentado en el Capítulo 3. una gran parte de los trabajos que están surgiendo sobre los TENGs se están centrando en el análisis de cómo pueden utilizarse para la captación y almacenamiento de energía (Jie, y otros, 2018) (Cheng, Gao, & Wang, 2019).

Este proyecto ha sido enfocado hacia el otro uso que se le está buscando a los TENG, como sensores y transmisores de información. Es vital que se siga desarrollando este aspecto de estos revolucionarios nano generadores, ya que en el futuro del IoT, uno de los retos clave es la alimentación de los sensores. Se persigue con el concepto de IoT un mundo en el que todo está conectado y se puede obtener información de todos los objetos y cosas gracias a sensores. Por este motivo si fuese necesario alimentar los sensores de cada objeto con baterías el gasto de energía sería inmenso y el aspecto económico no sería rentable en ningún caso. Sin embargo, con los TENGs surge la posibilidad de trabajar con sensores autoalimentados, proporcionando así una solución a uno de los mayores retos del IoT. Además, los TENGs cuentan con las ventajas de ser rentables, de estructura simple, de fácil fabricación (más fácil que los piezoeléctricos o PENG), portátiles y de alta fiabilidad. En este sentido se han desarrollado dos funcionalidades para la tecnología. Ambas son explicadas en la sección 4.2 como parte de los objetivos principales de este proyecto.

4.2 OBJETIVOS

En este proyecto se busca analizar la viabilidad y la fiabilidad de la transmisión de información de la tensión eléctrica generada por el TENG mediante sistemas de comunicación de IoT. Se persiguen tres objetivos que nacen de esta búsqueda:

- El primer objetivo, imprescindible para los dos siguientes, es la fabricación de un sistema de comunicación de protocolo IoT. Para poder transmitir cualquier información, antes hay que tener un canal a través del cual transmitirlo.
- El segundo objetivo es lograr la detección de cuando un el TENG es activado generando un pico de tensión. En este objetivo se busca simplemente la detección por software y la transmisión por IoT de un mensaje de alarma que alerte al usuario de la activación del pulso. Este objetivo se puede ligar a detectores de fuego, de movimiento, de lluvia y de cualquier fenómeno si se diseña el mecanismo adecuado para que se pulse el sensor.
- Un tercer y último objetivo es digitalizar la señal generada por el TENG, transmitirla y representar la información recibida. El análisis del pulso una vez recibido es crucial también, debido a que de nada sirve transmitirlo si en su recepción no se mantienen características de la señal como la frecuencia o la tensión alcanzada. De hecho, es vital que se conserven, porque gracias sobre todo a estas dos características podemos extraer información del suceso que ha provocado estos pulsos.

Metodología

Para la consecución de los objetivos mencionados en la sección 4.2 se propone la siguiente metodología. Una vez se ha elegido LoRaWAN como el protocolo de comunicación a emplear se debe estudiar que es necesario para transmitir información con este sistema (dispositivo final y gateway). Se necesita también analizar cómo se va a digitalizar la señal teniendo siempre en mente la frecuencia de pulsos. Será necesario el uso de máquinas que permitan generar pulsos con frecuencias altas para probar hasta que límites se puede llegar. Por último, se realizará un análisis de los pulsos enviados en comparación con los recibidos.

4.3 ESTIMACIÓN ECONÓMICA

Para la estimación económica se van a tener en cuenta solamente los costes directos de compra de placas que es realmente lo que puede suponer un coste importante. Por ejemplo, no se va a añadir el coste de los *jumper-wires* ni de las microSD. En la siguiente Tabla 1 se incluye los costes de cada placa por separado, así como el coste total del proyecto.

Placa	**Cantidad**	**Coste unitario (€)**	**Coste total (€)**
Arduino MKR WAN 1300	5	35	175
Raspberry Pi 2 Model B	1	43,95	43,95
Raspberry Pi 4 Model B	1	64	64
RAK831 LPWAN Gateway Concentrator Module	1	110,59	110,59
RAK2245 RPi HAT Edition	1	100,54	100,54
		TOTAL	494,08

Tabla 1. Tabla de costes directos del proyecto

Capítulo 5. SISTEMA/MODELO DESARROLLADO

5.1 NODOS FINALES

5.1.1 PLACA ARDUINO MKR WAN 1300

Como se ha introducido en la sección 2.2 para emplear el protocolo LoRaWAN es necesario un nodo final con tecnología LoRa que sea capaz de leer el pulso y enviarlo por comunicación inalámbrica. En el mercado hay numerosas placas que pueden realizar esta función, pero después de mucha investigación y por tratarse de una plataforma de desarrollo conocida y fiable se ha decidido elegir una placa de Arduino. En concreto la placa MKR WAN 1300 que se ilustra en la Figura 4.

Figura 4. Imagen de la placa Arduino MKR WAN 1300 (Arduino, 2021)

MKR WAN 1300 es una potente placa que combina la funcionalidad del MKR Zero y la conectividad LoRa / LoRaWAN. Se basa en el chip Atmel SAMD21 y en un módulo Lo-Ra Murata CMWX1ZZABZ. El cambio de una fuente a otra se realiza automáticamente. Esta placa cuenta con una buena potencia de cálculo de 32 bits (muy similar a la de las otras placas de Arduino como la MKR Zero), el habitual y rico conjunto de interfaces de E/S, la comunicación Lo-Ra de bajo consumo y la facilidad de uso del software Arduino (IDE) para el desarrollo de código y la programación (Arduino, 2021). Una de las ventajas

de la placa, por las que ha sido elegida, es que cuenta con entradas analógicas (A0-A6) que permiten realizar la conversión analógica-digital de la señal que se quiere transmitir. La placa se puede alimentar con dos pilas AA o AAA de 1,5V o con 5V externos. El cambio de las distintas fuentes de alimentación se realiza de manera automática (Store, 2021). En la siguiente Figura 5 observamos el diagrama de pines de la placa.

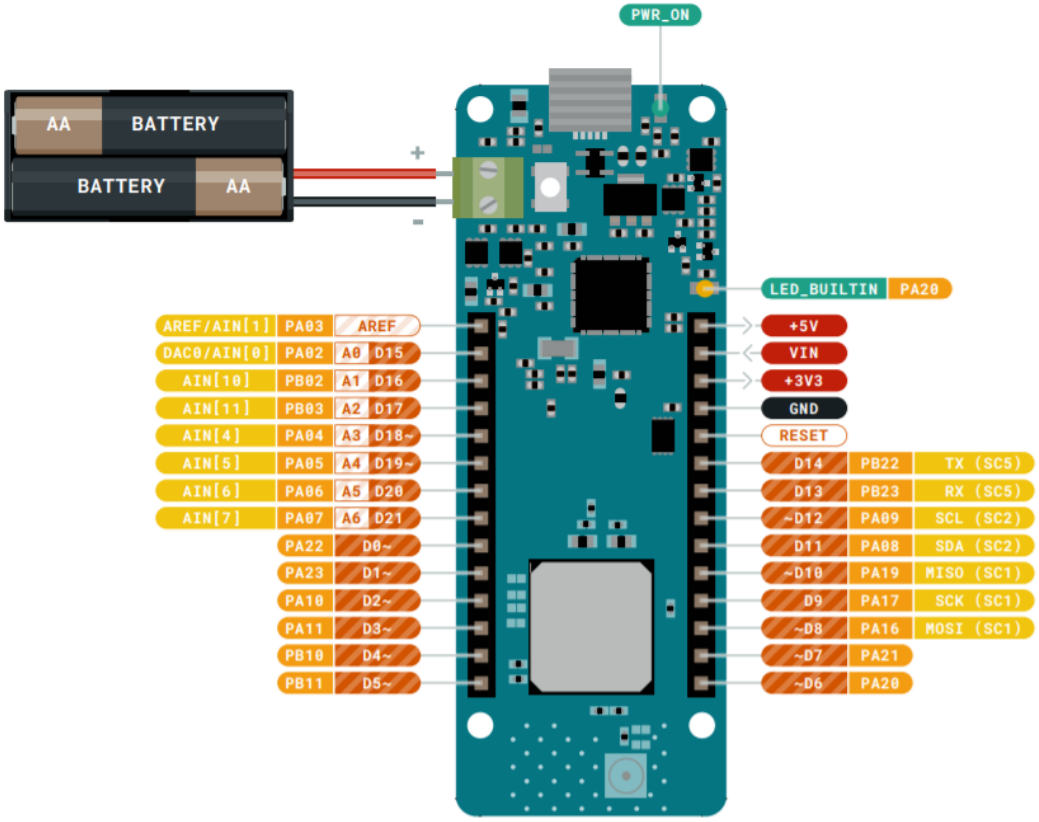


Figura 5. Diagrama de pines de la placa Arduino MKR WAN 1300 (Arduino, 2021)

Aunque la placa es capaz de emitir y recibir mensajes por sí sola, para potenciar su alcance se ha añadido una antena dipolo que viene representada en la Figura 6. Se trata de una antena dipolo GSM (*Global System for Mobile Communications)* con adhesivo y conector UFL. Las antenas dipolo se caracterizan por estas formadas por dos elementos. Uno de ellos es alimentado de forma directa mientras que el otro posee un acoplamiento inductivo en los extremos. Cada uno de los dos elementos cuenta con solo media longitud de onda de

largo, sin embargo, como las esquinas son capaces de conducir corriente se consigue una longitud de onda completa de corriente en la antena. Esta antena está sintonizada para funcionar en el rango GSM de frecuencias: 850/900/1800/1900 MHz. Como se ha comentado en la sección 2.2 en Europa el protocolo LoRaWAN trabaja con una frecuencia de 868 MHz por lo que cumple los requerimientos necesarios.

Figura 6. Imagen de la antena dipolo usada para aumentar el alcance de la comunicación

5.1.2 REGISTRO EN THE THINGS NETWORK

Para esta sección se ha incluido una serie de pantallazos ilustrativos en el ANEXO II que permiten una mejor comprensión de los pasos explicados en las distintas subsecciones.

5.1.2.1 Creación de aplicación

Como se ha comentado en la sección 2.3 es necesario crearse una cuenta en The Things Network y crear una aplicación donde poder registrar los dispositivos finales que transmitirán la información. Crearse una cuenta puede hacerse desde la página web https://www.thethingsnetwork.org/ de manera trivial con una cuenta de email.

Una vez creada la cuenta se debe clicar en "*console"* y luego en "*applications"* para poder generar una nueva aplicación. El siguiente paso es clicar en "*add application"* que lleva a un menú en el que se fijan las características de la aplicación. Lo primero que hay que fijar es el "*application ID"* que es el identificador único que diferencia las aplicaciones entre sí dentro del servidor de red. Luego se añade la "*description"* que, como su propio nombre indica, es la descripción de la aplicación que se está creando. El siguiente campo, "*application EUI (Extended User Interface)"*, es auto rellenado por The Things Network. Se trata de un identificador único de 64 bits que es asignado por el servidor de cuentas de

The Things Network desde el bloque de direcciones de The Things Network Foundation en el momento de creación de una aplicación. Para finalizar, en el último campo, "*handler registration"*, se ha de seleccionar el *handler* para nuestra aplicación. Un *handler* se encarga de gestionar los datos de una o varias aplicaciones. Para ello, se conecta a un *broker* (el bróker es la columna vertebral global del tráfico LoRaWAN que permite el intercambio de forma segura entre las distintas redes) donde registra las aplicaciones y los dispositivos. El *handler* realiza también el encriptado y desencriptado de datos (TheThingsNetworkNetworkArchitecture, 2021). Existe la posibilidad de configurar un *handler* privado que se conecte a la red pública comunitaria como explica Hylke Visser uno de los fundadores de The Things Network (Visser, 2021). Una vez rellenados todos los campos y tras clicar en *Add application* la aplicación ya está creada y se puede comprobar el *application EUI* que es necesario cuando se programa el MKR WAN 1300. Otro campo importante es el *access key* que es muy importante a la hora de almacenar y acceder a la información transmitida.

5.1.2.2 Registrar dispositivo

Una vez creada la aplicación ya se puede registrar los MKR WAN 1300 como dispositivos asociados a esta aplicación. Para ello es necesario extraer de cada placa un código identificador único o *device EUI* mediante un código previamente programado por Gonzalo Casas que se encuentra para uso público en GitHub y se incluye en el ANEXO I al final de este documento. Este código fuente se encuentra bajo el nombre "mkrwan_01_get_deveui". Tras conectar la placa al ordenador mediante un cable USB 2.0 Micro-B 5 Pin se realiza la comunicación con la placa mediante el software de Arduino. En "Herramientas" → "Placa" → "Gestor de tarjetas" se debe instalar el paquete que incluye la tarjeta MKR WAN 1300 que se llama "Arduino SAMD Boards (32-bits ARM Cortex-M0+)". También es necesario instalar la librería de esta placa en "Programa" → "Incluir librería" → "Administrar bibliotecas…" bajo el nombre de MKRWAN. Una vez descargadas la librería y la tarjeta el software de Arduino ya debería reconocer la placa en uno de los puertos COM del ordenador con el que se esté trabajando. El último paso para obtener el *device EUI* es compilar el código "mkrwan_01_get_deveui" y por el monitor

serie aparecerá un mensaje como el de la Figura 7 en el que queda recogido el *device EUI* del MKR WAN 1300 conectado.

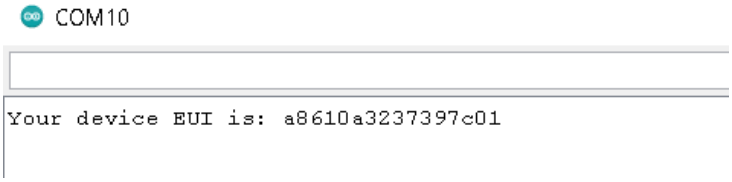


Figura 7. Monitor serie al compilar el código "mkrwan_01_get_deveui"

Una vez se ha extraído este código de la placa ya se puede registrarla dentro de la aplicación. En el menú *overview* de la aplicación se clica en *+register device* y aparecerá un menú con cuatro campos que rellenar. El primero bajo el nombre de *Device ID* es muy importante ya que este será el nombre que permita diferenciar entre los distintos nodos finales a los que estarían conectados los TENG. Por ejemplo, si se utiliza la funcionalidad de alarma, en un gran número de motores para detectar si se produce el desprendimiento de algún elemento, que active el TENG conectado a un MKR WAN 1300; es importante que desde el centro de control que monitorice el estado de los motores se pueda identificar qué dispositivo es el que ha sufrido la anomalía cuando se reciba el mensaje de alarma. El siguiente campo para rellenar es el *Device EUI* que se ha obtenido de la placa mediante el código "mkrwan_01_get_deveui". Los dos últimos campos, *App key* y *App EUI*, son auto rellenados y son iguales para todos los dispositivos de una misma aplicación. No son otra cosa que la *App key* y *App EUI* que ha generado The Things Network cuando se ha creado la aplicación que se explica en la sección 5.1.2.1. Tras clicar en *Register* el dispositivo ya se encuentra registrado y listo para transmitir y recibir información.

5.1.2.3 Decodificadores

Otro aspecto muy importante de la configuración de la aplicación es la decodificación de la información para que esta pueda entenderse de manera fácil y rápida. Este proceso se realiza mediante código dentro del menú de aplicación en el apartado de "*Payload Formats"*, en concreto en "*decoder"*. Aquí se inserta un código de decodificación que traduce de un lenguaje hexadecimal al lenguaje con el que se haya enviado el mensaje.

Es importante remarcar que en esta sección no se produce el desencriptado de la información. El encriptado es realizado antes del envío del mensaje y desencriptado en su recepción por la propia red de The Things Network. Es decir, cuando se decodifica el mensaje, este ya ha sido desencriptado y transformado a lenguaje hexadecimal. La decodificación se realiza porque este lenguaje es poco intuitivo de comprender.

Para programar un código de decodificación es importante saber primero qué se está enviando y en qué formato se están enviando la información desde el dispositivo final. En este proyecto se han programado dos decodificadores distintos para cada una de las dos funcionalidades a emplear:

- En un primer caso el decodificador simplemente se encarga de traducir el texto de alarma. En este caso el mensaje que se envía desde el MKR WAN 1300 se encuentra en formato ASCII 244 (texto legible). Realmente, este formato no es recomendable para la transmisión de información por LoRaWAN ya que ocupa mucho ancho de banda y este es limitado. Sin embargo, como el objetivo es simplemente transmitir el estado del dispositivo de alarma con una palabra ya sea seguro *(safe)* o peligro *(danger)*, no alcanzamos valores superiores a los límites de ancho de banda. Para hacer esta conversión se usa la función `String.fromCharCode.apply(null, bytes)` donde *bytes* es el mensaje recibido. En la siguiente Figura 8 se puede observar cómo en el primer envío que se ha realizado sin decodificador (16:36:41) el *payload*, que no es otra cosa que el mensaje transmitido, no es legible. Por otro lado, en el segundo envío (16:37:06) que sí se ha realizado con el decodificador mencionado, rápidamente se puede identificar que se está informando del estado de la alarma, siendo este *DANGER* (PELIGRO).

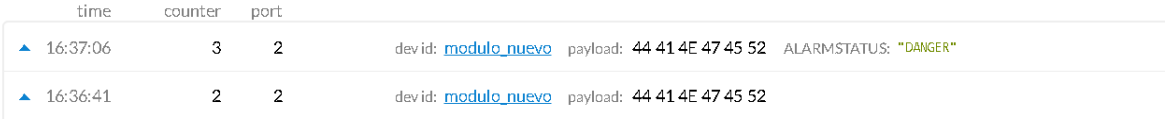


Figura 8. Diferencias entre envío con y sin decodificador texto ASCII 244

- En el segundo caso el decodificador debe transformar los bytes enviados en valores decimales para que se pueda representar la señal completa. El *payload* o mensaje enviado se transmite como un vector por lo que hay que decodificar cada uno de sus elementos que han sido enviados como bytes. Enviar la información como bytes es la mejor forma para maximizar el envío de información ya que es el formato que menos ancho de banda utiliza. Por tanto, la conversión necesaria para cada elemento del vector mensaje es de un byte en hexadecimal a un número decimal. En las siguientes imágenes se ilustra la importancia de que cada decodificador tiene que estar diseñado para el objetivo que se persigue. Por ejemplo, en la Figura 9 se ha probado el decodificador de hexadecimal a texto ASCII 244 mencionado en el punto anterior para un mensaje que contiene un pulso digitalizado. Como es de esperar, la decodificación no tiene ningún sentido y se observa una serie de caracteres de texto sin ningún significado. Sin embargo, en la Figura 10 se ha empleado el decodificador de hexadecimal a decimal mencionado en este punto, y el resultado obtenido muestra el periodo del pulso y los valores (escalados) del propio pulso digitalizado para su futura representación (este proceso se explicará más adelante en la sección 5.3.2).

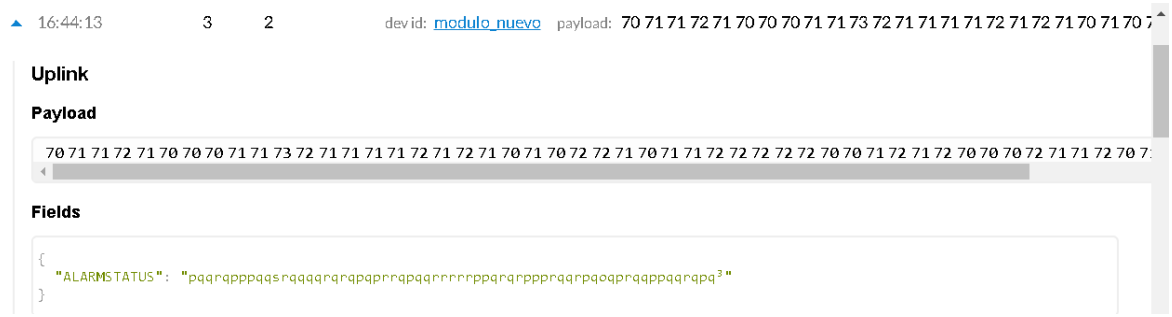


Figura 9. Ejemplo de decodificador de texto ASCII 244 para transmisión de pulso digitalizado

Figura 10. Ejemplo de decodificador decimal para transmisión de pulso digitalizado

5.1.2.4 Visualización de datos

Mientras se está transmitiendo información, es importante que se disponga de una forma de visualizar la información en tiempo real. Se puede acceder a ella desde el menú de aplicación en *Data*. Esta función permite acceder al mensaje enviado desde cualquier parte del mundo, siempre que se tenga conexión a internet. Esto supone que se puede monitorizar en tiempo real cualquier dispositivo final que esté vinculado a la aplicación sin necesidad de encontrarse si quiera en el mismo continente que las placas emisoras. Gracias a esta posibilidad el abanico de aplicaciones se vuelve inmenso pudiendo tener controlados distintos objetos colocados en distintas partes del mundo bajo un mismo centro de control que se localice en la sede central de la compañía.

La información que se proporciona en el mensaje es además muy extensa como se puede comprobar en la Figura 11. En ella se puede observar la hora los minutos y los segundos a la que se ha enviado ese mensaje en la columna *time*. En la columna *counter* se encuentra el número de mensajes enviados desde que el dispositivo final ha establecido contacto con un gateway conectado a The Things Network. La sección de *port* simplemente hace referencia al gateway por el que está circulando el tráfico de información. Los módulos asignan un 1 al primer gateway al que se conectan, un 2 al segundo y así sucesivamente. El *dev id* es uno de los puntos más importantes ya que es lo que permite diferencia tus dispositivos finales entre sí dentro de la aplicación. En *payload* se puede observar el mensaje en hexadecimal tal y como ha sido enviado y antes de ser decodificado y a su derecha el mensaje decodificado. *Uplink* hace referencia al tipo de mensaje enviado, desde

el dispositivo final al servidor de aplicación (si fuese desde la aplicación al dispositivo aparecería *downlink* en su lugar). En *Metadata* aparece el día y la hora de transmisión mucho más precisa y en referencia UTC (Universal Time Coordinated), la frecuencia de onda que se ha transmitido en MHz, el tipo de modulación y más metainformación que carece de relevancia para este proyecto. En la parte de abajo se encuentra el "*estimated airtime"* que quiere decir el tiempo de vuelo que ha experimentado el mensaje. Este tiempo aumenta cuanto mayor sea el tamaño del mensaje.

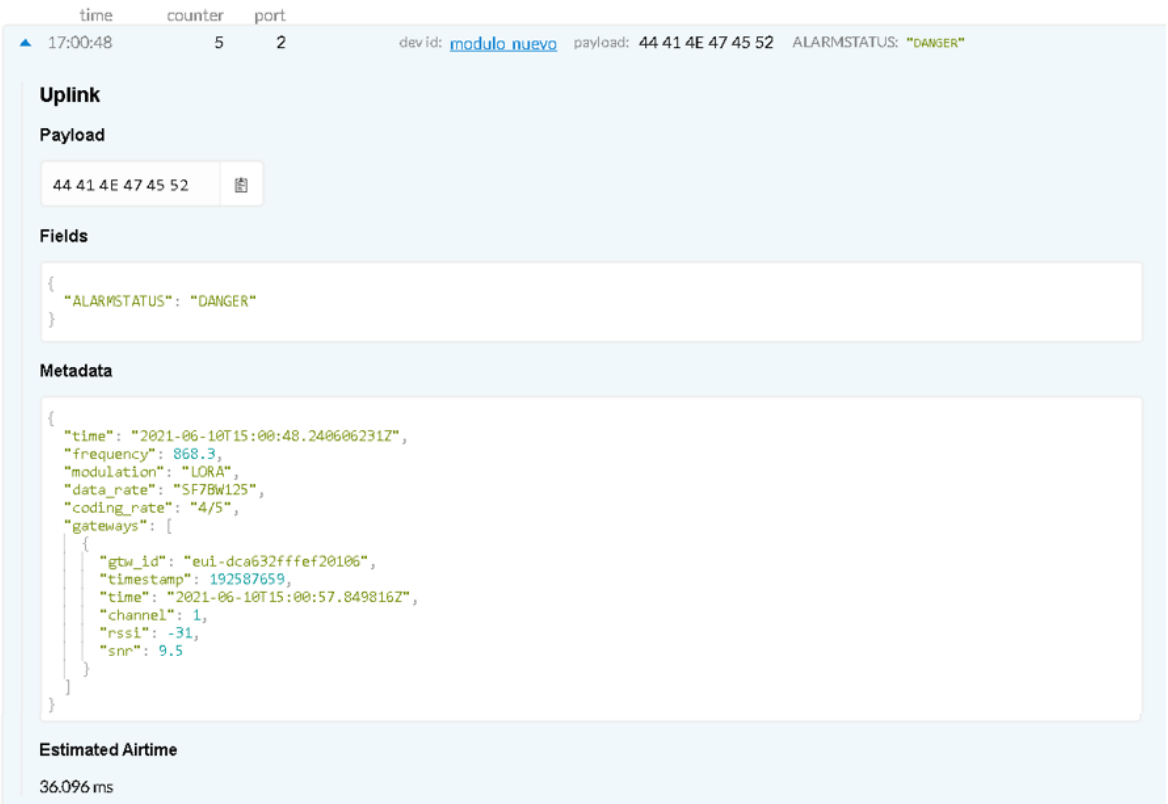


Figura 11. Ejemplo de la información acerca de un mensaje transmitido

5.1.2.5 Almacenamiento de datos

Aunque la función anterior tiene las ventajas de ser accesible de manera rápida y permitir un seguimiento en tiempo real, no permite almacenar la información ya que esta desaparece si se recarga la página o se cierra el navegador. Otro problema que plantea es que no permite la selección de todos los valores recibidos cuando el mensaje es un vector

de gran tamaño. Para la representación del pulso digitalizado con una herramienta de procesamiento de datos como Excel, esto incomoda mucho la tarea obligando al usuario a copiar y pegar varias veces por cada mensaje recibido. Para solucionar estas dos dificultades existe más de una alternativa, cada una con aspectos positivos y negativos. Una opción es utilizar la integración de The Things Network con Ubidots. Aunque esta opción está más orientada a almacenar solo uno o dos datos y en intervalos muy grandes de tiempo (por ejemplo, temperatura y humedad de una instalación en intervalos de una hora), podría resultar una alternativa cómoda para el registro de los datos y la elaboración de gráficas. Sin embargo, Ubidots exige una suscripción de pago, y por este motivo se ha decidido investigar otros métodos quizás más enrevesados, pero más económicos. Otra opción, a priori la más adecuada, es una integración de tipo HTTP que permite almacenar los datos en una *spreadsheet* de Google Docs. El problema es que existen algunos fallos en The Things Network que han imposibilitado trabajar con esta integración; en futuras versiones que se solucionen se recomienda seguir esta vía. Finalmente se ha decidido utilizar la integración de Data Storage propia de The Things Network aún con las limitaciones que esta conlleva. Esta integración almacena automáticamente toda la información transmitida por un periodo de 7 días. Una vez pasado este periodo, la información se borra, lo que supondría un problema serie si se quisiera almacenar en una base de datos para registros históricos. Otro de los problemas es que la información se almacena por mensajes en formato JSON. Este en cambio no es muy grave ya que se puede copiar y pegar en Excel fácilmente para representar el pulso. Claramente no supone la alternativa más práctica ni la más cómoda, sin embargo, para los distintos experimentos de los que requiere este trabajo es más que suficiente. Para acceder a esta integración simplemente hay que activarla en *Integrations* e introducir la llave de acceso de la aplicación o *App key* que se ha mencionado en el apartado 5.1.2.1.

5.1.3 MONTAJE FINAL

En la siguiente Figura 12 se muestra el esquemático del módulo MKR WAN 1300 para las dos funcionalidades que se explican en la sección 5.3. que. En el esquemático se conecta

un TENG mediante dos conectores de plástico entre tierra y el pin A1, pero valdría cualquier pin de entrada analógica de la placa. También se adjunta una imagen real del montaje terminado en la Figura 13.

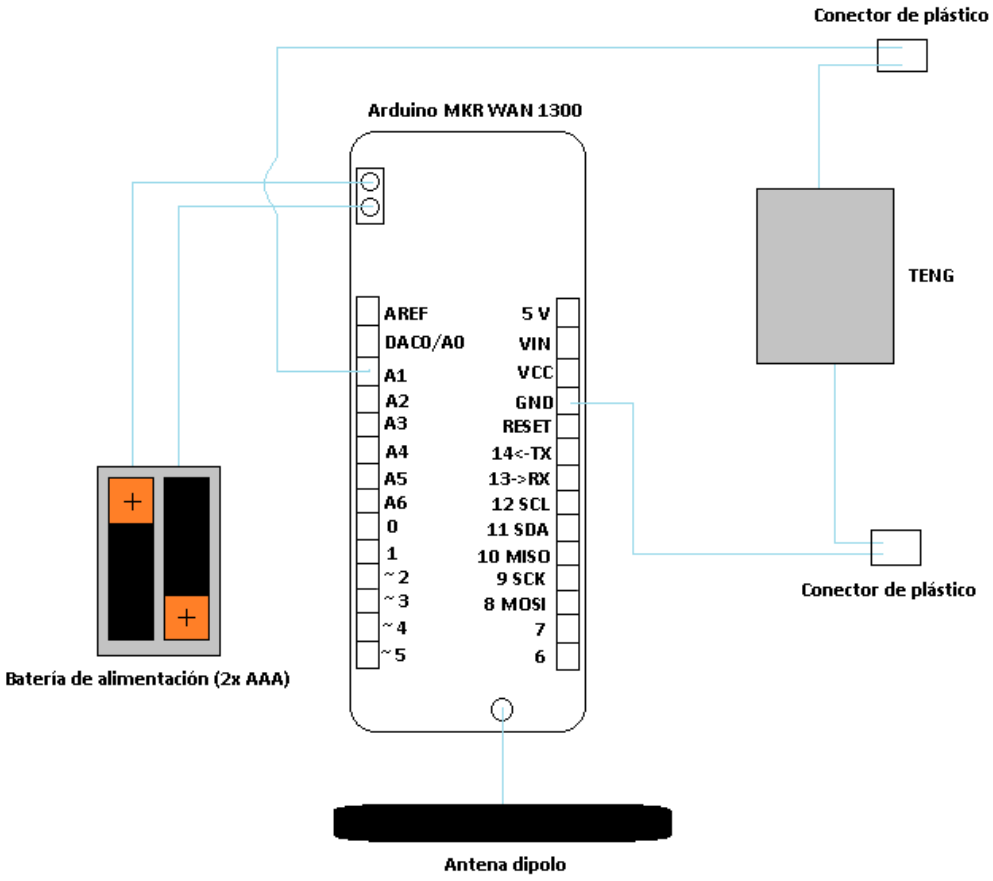


Figura 12. Esquemático final Arduino MKR WAN 1300

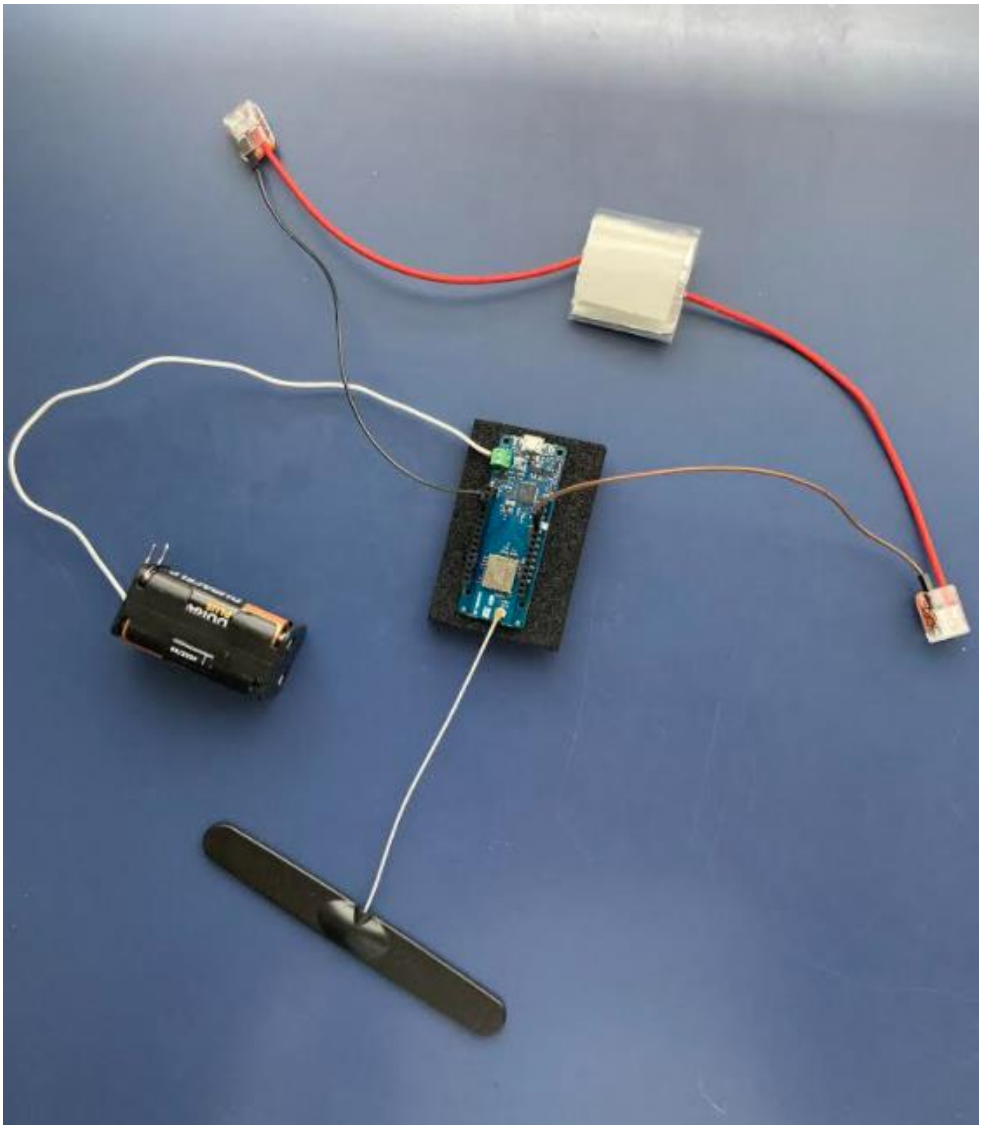


Figura 13. Imagen del montaje terminado del Arduino MKR WAN 1300

5.2 GATEWAYS

Para que el mensaje transmitido por la placa de Arduino sea recibido y subido a internet es necesaria que una pasarela o gateway se encargue de esto. La comunidad de LoRa Alliance crece cada día y cada vez más gateways se registran en The Things Network de uso público. Sin embargo, para poder hacer el mayor número de pruebas sin tener que desplazarse, se decidió que era más adecuado contar con un gateway propio que permitiese el trabajo desde IMDEA Materiales o desde la escuela ICAI. Una opción que existe es la de comprar uno ya programado y montado como pueden ser los que desarrolla la empresa RAK. Sin embargo, por motivos económicos (los gateways se venden por precios de entre 300 € y 1000 €) y didácticos se decidió programar de manera personal el gateway. Para montar un gateway es necesario una placa que tenga un módulo transceptor de comunicación LoRa y un ordenador que haga de host para esta placa. En este apartado se explican los procesos de montaje de dos gateways: uno con comunicación inalámbrica (wifi) a internet y el otro por comunicación por cable (ethernet) a internet.

5.2.1 GATEWAY CON MÓDULO RAK 831 Y CONEXIÓN ETHERNET

5.2.1.1 RAK 831 y Raspberry Pi 2 Model B

Para este gateway se ha seleccionado una de las combinaciones de mejor rendimiento del mercado. Como host se ha utilizado una Raspberry Pi 2 Modelo B (Figura 15) y como módulo concentrador la placa RAK 831 (Figura 14). La empresa RAK es una empresa de origen chino que se dedica al desarrollo de bloques de construcción de IoT orientados al mercado y fáciles de usar de MCU, módulos de comunicación y sensores en un sistema de plataforma modular de IoT. Uno de los protocolos que más está siendo desarrollado en RAK es el protocolo LoRaWAN por lo que se trata de una de las empresas punteras en la producción de gateways funcionales y módulos concentradores para su fabricación manual. El módulo RAK 831 incluye una antena de transmisión LoRa.

Figura 14. Imagen del módulo concentrador RAK 831 (RAKWireless, 2021)

Figura 15. Imagen de la placa Raspberry Pi 2 Model B (RaspberryPi, 2021)

5.2.1.2 Configuración del gateway

Para programar este gateway es necesario configurar primero la Raspberry Pi cargando en ella el sistema operativo Raspberry Pi OS Lite que se puede descargar en la propia página de Raspberry Pi siguiendo el siguiente enlace. Para cargar este sistema operativo se utiliza una microSD previamente formateada con SD Card Formatter. Hay más de un programa de descarga gratuita que permite cargar el sistema operativo de Raspberry a una tarjeta microSD. En este trabajo se ha probado balenaEtcher y Win32DiskImager obteniéndose con ambos resultados satisfactorios.

Antes de conectar la Raspberry Pi a la fuente de alimentación es necesario introducir en ella la microSD con el sistema operativo cargado y conectarla al módulo RAK 831 mediante cables *jumper-wire* macho para el módulo RAK 831 y hembra para la Raspberry Pi. La conexión se hace siguiendo la siguiente Tabla 2.

Pin RAK 831	Descripción	Pin Raspberry Pi
1	+ 5 V	2
3	GND (tierra)	6
15	CSN (Chip Select)	24
16	MOSI	19
17	MISO	21
18	SCK (SPI Clock)	23
19	RST (Reset)	22

Tabla 2. Guía de conexión de pines entre RAK 831 y Raspberry Pi 2 Model B

Una vez realizada la conexión de ambas placas, ya se puede encender la Raspberry Pi conectándola a una fuente de alimentación de 5 V y 3 A.

Una opción para acceder a la interfaz del sistema operativo de la Raspberry Pi es conectarle una pantalla por la entrada HDMI que tiene disponible y por los puertos USB conectarle un ratón y un teclado. Sin embargo, en este trabajo se ha accedido a la interfaz por SSH ya que esta opción es más simple y menos aparatosa. Para que sea posible la conexión con el ordenador vía SSH es imprescindible conectar la Raspberry Pi a internet mediante un cable ethernet. También es necesario incluir un archivo llamado *ssh* sin extensión en la partición boot de la microSD para poder realizar conexiones SSH con la Raspberry Pi, ya que esta opción viene deshabilitada por defecto. El servidor DHCP (*Dynamic Host Configuration Protocol*) del router al que está conectado nuestro cable

ethernet asigna una dirección IP dinámica a la Raspberry Pi. Para poder conectarse, hay que localizar que IP le ha sido asignada. Para ello es necesario un programa como Advanced IP Scanner. Tras darle al botón de explorar, este programa muestra todas las direcciones IP que hay asignadas en la red wifi a la que está conectada el ordenador. Cada dirección aparece con su fabricante por lo que es muy simple identificar cual es la dirección IP que ha sido asignada a nuestra placa Raspberry Pi.

Para conectarse por SSH se debe utilizar un programa habilitado para ello como puede ser Putty. Simplemente basta con introducir la dirección IP obtenida de Advanced IP Scanner y el ordenador se conecta con la interfaz de Raspberry Pi inmediatamente. Dentro de ésta, se pide un usuario y contraseña. Por defecto, el usuario es "pi" y la contraseña "raspberry". Una vez dentro, es recomendable modificar esta contraseña para aumentar la seguridad del dispositivo.

El comando que hay que introducir para acceder a la configuración de la Raspberry Pi es `sudo raspi-config`. Al introducir este comando se abre un menú en el que se deben de realizar varias modificaciones. La primera de ellas es el cambio de la contraseña en "*1 Change User Password*". A continuación, es necesario configurar la localización en "*4 Localisation Options*" → "*I1 Change Locale*". Primero se debe cargar la zona española que es "es_ES.UTF-8 UTF-8" y posteriormente seleccionarla como zona por defecto. Es importante también configurar las conexiones periféricas en "*5 Interfacing Options*". Se debe habilitar el protocolo SSH si aún no se ha habilitado en "*P2 SSH*". Como la comunicación con el módulo radio RAK 831 se va a realizar mediante SPI (*Serial Peripherial Interface*) es fundamental su activación también en "*P4 SPI*" para el correcto funcionamiento del gateway. Tras realizar esta activación ya se puede finalizar el proceso de configuración y reiniciar la placa para que se apliquen todos los cambios con el comando `sudo reboot`. Este comando termina la comunicación entre el ordenador y la Raspberry Pi así que para continuar con la programación hay que volver a conectarlos con Putty siguiendo el mismo proceso usado anteriormente. Una vez se ha realizado la conexión de nuevo y se ha introducido el nuevo usuario y contraseña hay que actualizar el sistema con los comandos `sudo apt-get update` y `sudo apt-get upgrade` en ese orden.

Para clonar el software del repositorio git e instalarlo en la Raspberry Pi es importante su instalación con el comando `sudo apt-get install git`. Una vez instalado se clona el repositorio donde se encuentra la aplicación para instalar con el comando `git clone -b spi https://github.com/ttn-zh/ic880a-gateway.git ~/ic880a-gateway`. Una vez se ha clonado se accede al directorio donde se ha descargado la aplicación con `cd ~/ic880a-gateway` donde se pueden observar los distintos archivos que existen. Se puede comprobar las características de la aplicación accediendo al archivo de inicio con `sudo nano start.sh`. El último paso es la instalación de The Things Network con `sudo ./install.sh spi`. Una vez ejecutado este comando es importante apuntar el Gateway EUI que se detecta ya que será vital para el registro del gateway en The Things Network. Al ejecutar este comando se pregunta si se quiere habilitar la configuración remota a través de un archivo. Si se quisiera habilitar habría que crear un archivo json y subirlo al repositorio, hacer un *for* y solicitar que s e incluya en el repositorio principal ese archivo. Se ha decidido no habilitarla para simplificar la instalación, por lo que se creará un archivo json localmente y cada vez que se inicie el módulo se accederá a este archivo de configuración con los parámetros que se establecen a continuación. Los parámetros que hay que configurar son el nombre, una pequeña descripción, el mail de contacto y la localización del gateway en latitud, longitud y altura. Tras finalizarse el proceso de instalación se ha terminado la configuración del gateway y se puede proceder a su registro en The Things Network.

5.2.1.3 Registro en The Things Network

Al igual que para la sección 5.1.2 se han incluido un número de pantallazos ilustrativos en el ANEXO II de manera que se facilite la comprensión de los pasos necesarios para el registro de un gateway en The Things Network.

Para registrar el gateway en The Things Network se debe acceder a *Console* en la cuenta y clicar en *Gateways*. Aquí se selecciona el botón *+register device* y aparecerá un menú con varios campos que rellenar. El primero es el *Gateway ID* donde se debe introducir el Gateway EUI que se ha obtenido en la sección 5.2.1.2. Se debe marcar la casilla de uso del reenviador de paquetes o *legacy packet forward*. En *Description* se puede incluir una descripción del gateway incluyendo tipo, uso, modelo o todo aquello que se crea

conveniente. El *Frequency Plan* que se elija depende de la zona donde se vaya a trabajar con ese *gateway* siendo Europa 868 MHz en el caso concreto de este trabajo. El campo de *Router* se rellena automáticamente al seleccionar el plan de frecuencias. Se puede marcar la localización del gateway y concretar si este va a ser de uso interior o exterior (interior en este trabajo). Tras clicar en *Register gateway* el gateway ya se encuentra operativo y siempre que se alimente la Raspberry Pi y se conecte a internet con un cable ethernet se podrán transmitir mensajes a través del mismo. Se puede comprobar que el gateway está conectado cuando aparece el indicador *connected* en el campo *status* como aparece en la Figura 16.

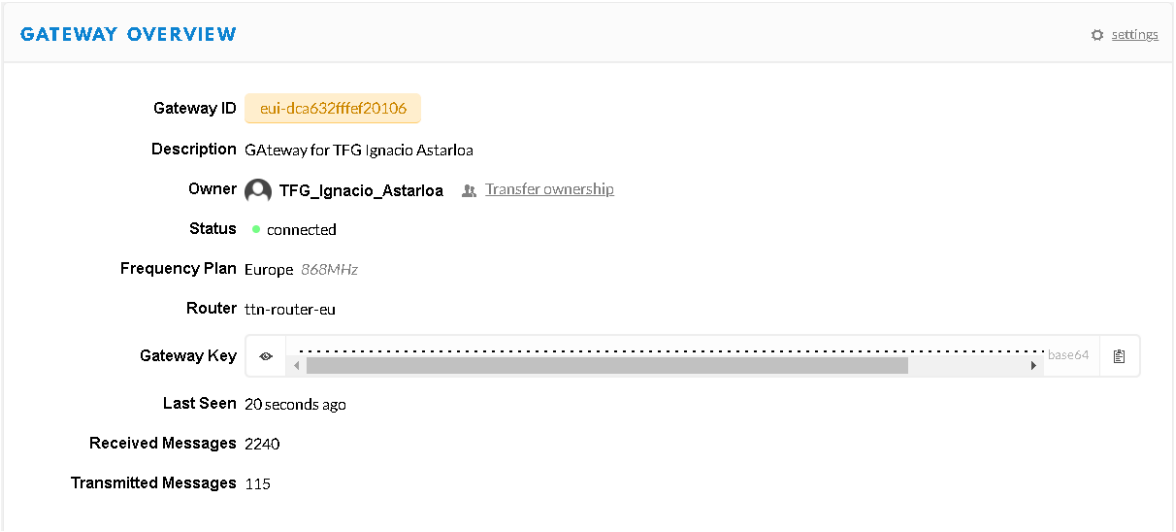


Figura 16. Ejemplo de un gateway y registrado y en funcionamiento

5.2.1.4 Montaje final

En la siguiente Figura 17 se muestra el esquemático del módulo gateway con el módulo RAK 831 y conexión por ethernet. También se adjunta en la Figura 18 una imagen del montaje real en funcionamiento.

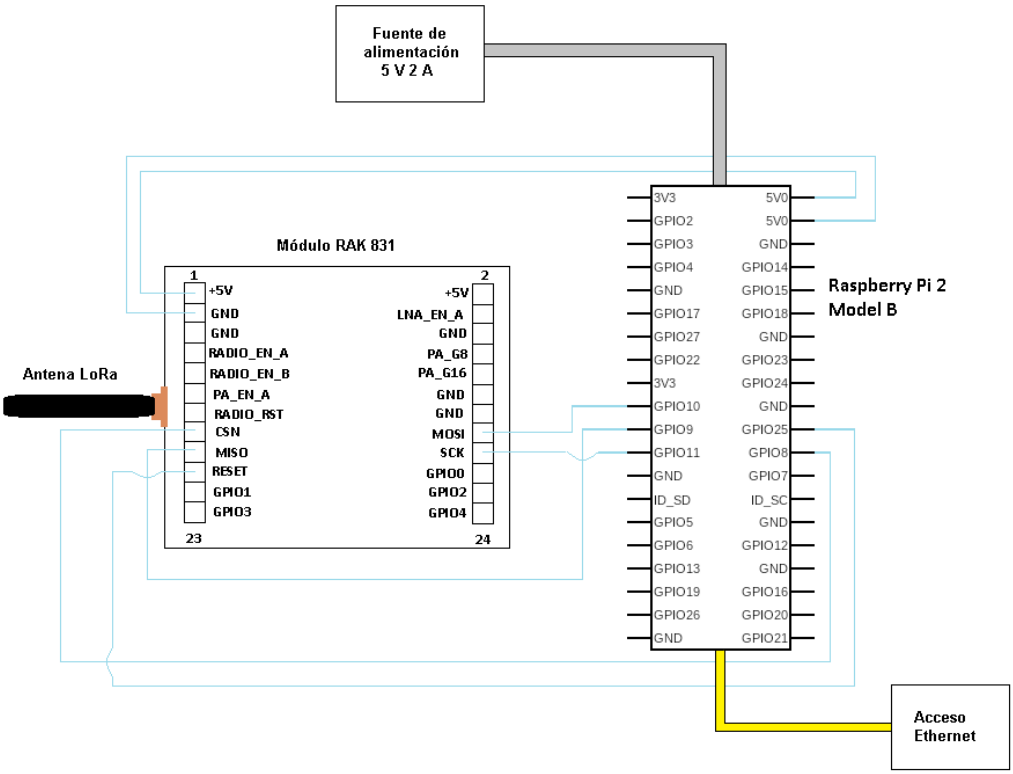


Figura 17. Esquemático del gateway con módulo RAK 831

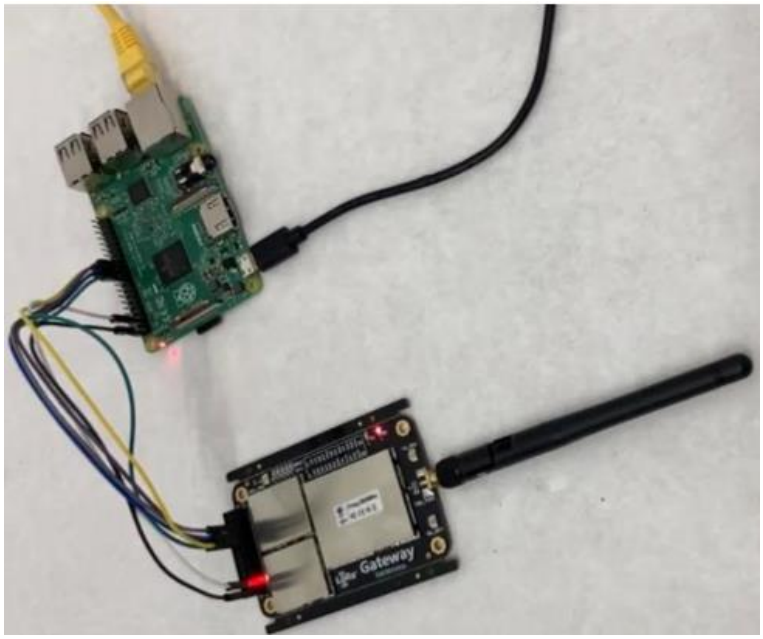


Figura 18. Imagen del montaje del gateway con módulo RAK 831

5.2.2 GATEWAY CON MÓDULO RAK 2245 Y CONEXIÓN WIFI

5.2.2.1 RAK 2245 y Raspberry Pi 4 Model B

Para el segundo gateway se ha vuelto a elegir una combinación de la empresa RAK con Raspberry Pi. En este caso el módulo concentrador es más moderno y está diseñado para tener una mayor facilidad en la configuración y montaje del gateway. Se trata del módulo RAK2245 RPi HAT Edition y se puede observar en la Figura 19. Al tratarse de un módulo más moderno el ordenador que hace de host debe ser más moderno también por lo que se ha seleccionado la Raspberry Pi 4 Model B que se ilustra en la Figura 20. El módulo RAK 2245 incluye una antena de transmisión LoRa y una antena de GPS.

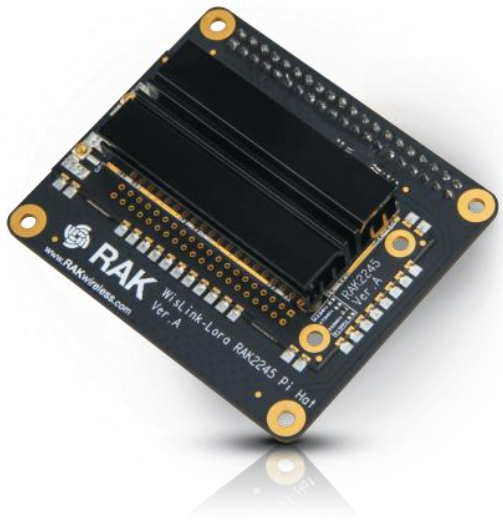


Figura 19. Imagen del módulo concentrador RAK2245 RPi HAT Edition (RAKWireless, 2021)

Figura 20. Imagen de la placa Raspberry Pi 4 Model B (RaspberryPi, 2021)

5.2.2.2 Configuración del gateway

La configuración de este gateway es más simple que la del anterior ya que el módulo está completamente orientado a funcionar con una Raspberry Pi. De hecho, la disposición de pines está diseñada para que se monten encima de los pines de la Raspberry Pi sin necesidad de jumper-wires como se ilustra en la siguiente Figura 21.

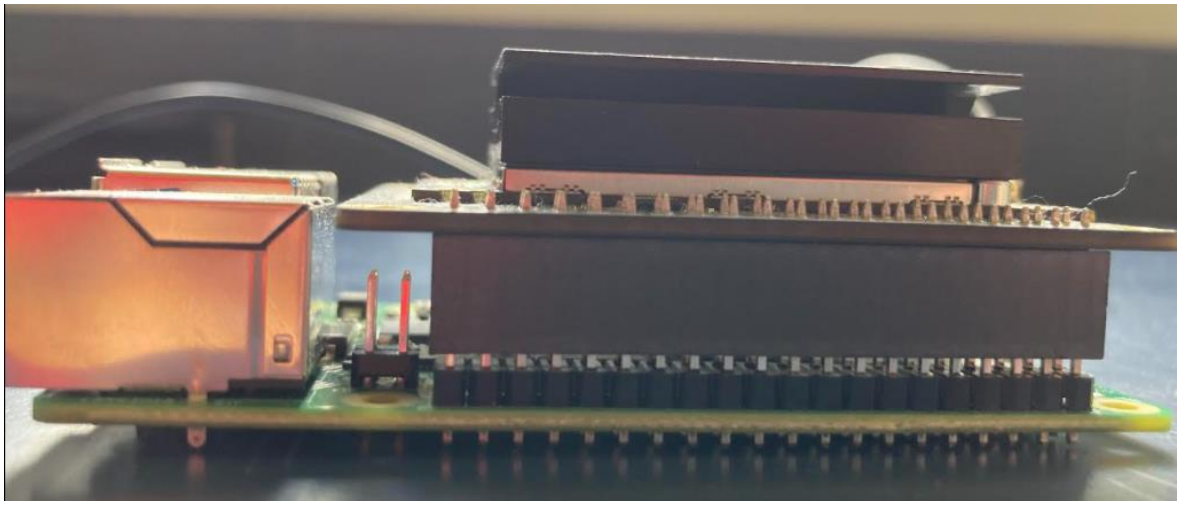


Figura 21. Imagen de la conexión de la Raspberry Pi y el módulo RAK 2245

Similarmente al gateway anterior hay que comenzar por cargar el sistema operativo de RAK 2245 para Raspberry 4 que podemos encontrar en downloads.rakwireless.com. Como en el gateway anterior, para cargar este sistema operativo se utiliza una microSD previamente formateada con SD Card Formatter. Como ya se ha mencionado se puede

cargar el sistema operativo con balenaEtcher o con Win32DiskImager en la microSD. Una vez introducida la tarjeta en la Raspberry Pi y con el módulo radio montado sobre sus pines se puede alimentar las placas con una fuente de alimentación USB-C de 3A.

Al alimentarse las placas se va a generar un punto de acceso wifi con el nombre de Rakwireless_XXX. Con el ordenador hay que conectarse a este punto de acceso y establecer comunicación con la Raspberry Pi por SSH. Con el programa ya utilizado Putty, hay que conectarse a la dirección IP 192.168.230.1 para establecer esta comunicación. Tras establecer la comunicación hay que introducir l usuario y contraseña. Como en el caso anterior por defecto son "pi" y "raspberry" respectivamente.

En este gateway hay que realizar algunas configuraciones del sistema también antes de que pueda utilizarse. Para acceder a la configuración se ejecuta el comando `sudo gateway-config` que genera un menú con distintas opciones de configuración. En "*1 Set pi password*" se puede cambiar la contraseña de acceso, lo qué es recomendable para aumentar la seguridad. En "*2 Setup RAK Gateway Lora concentrator*" se debe asignar el plan de servidor o *server plan* a TTN (The Things Network) y el plan de canal o *channel plan* al de la zona en la que vaya a funcionar ese gateway; en este trabajo Europa así que se selecciona EU_863_870. Como en este gateway se va a trabajar con conexión a internet wifi hay que continuar en el campo "*Configure WIFI*". Aquí lo primero que hay que hacer es habilitar el modo cliente en "*2 Enable client mode/Disable AP Mode".* Una vez habilitados ya se puede añadir el SSID y la contraseña de la red wifi a la que se va a conectar el gateway en "*4 Add new SSID for client*". Tras añadirla, se puede cerrar el menú de configuración y ejecutar el comando `sudo gateway-version` para obtener el gateway EUI que se necesita para su registro en The Things Network. Por último, solo queda reiniciar la Raspberry Pi con `sudo reboot` y automáticamente se desconectará el ordenador del acceso wifi.

5.2.2.3 Registro en The Things Network

Esta sección es idéntica a la 35 pero se incluye de nuevo para evitar confusiones. También existe un apartado en el ANEXO II para poder seguir con mayor facilidad el contenido de la sección.

Para registrar el gateway en The Things Network se debe acceder a *Console* en la cuenta y clicar en *Gateways*. Aquí se selecciona el botón *+register device* y aparecerá un menú con varios campos que rellenar. El primero es el *Gateway ID* donde se debe introducir el Gateway EUI que se ha obtenido en la sección 5.2.2.2. Se debe marcar la casilla de uso del reenviador de paquetes o *legacy packet forward*. En *Description* se puede incluir una descripción del gateway incluyendo tipo, uso, modelo o todo aquello que se crea conveniente. El *Frequency Plan* que se elija depende de la zona donde se vaya a trabajar con ese *gateway* siendo Europa 868 MHz en el caso concreto de este trabajo. El campo de *Router* se rellena automáticamente al seleccionar el plan de frecuencias. Se puede marcar la localización del gateway y concretar si este va a ser de uso interior o exterior (interior en este trabajo). Tras clicar en *Register gateway* el gateway ya se encuentra operativo y siempre que se alimente la Raspberry Pi y se conecte a internet con un cable ethernet se podrán transmitir mensajes a través del mismo. Se puede comprobar que el gateway está conectado cuando aparece el indicador *connected* en el campo *status* como aparece en la Figura 22.

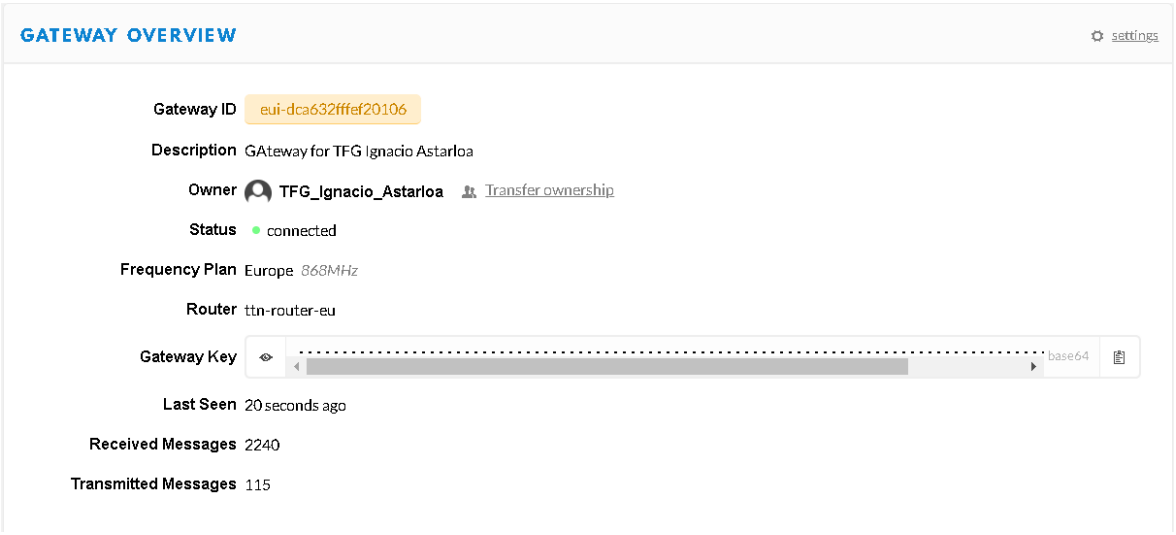


Figura 22. Ejemplo de un gateway y registrado y en funcionamiento

5.2.2.4 Montaje final

En la siguiente Figura 24 se muestra el esquemático del módulo gateway con el módulo RAK 2245 y conexión por ethernet. También se adjunta en la una imagen del montaje real en funcionamiento en la Figura 23.

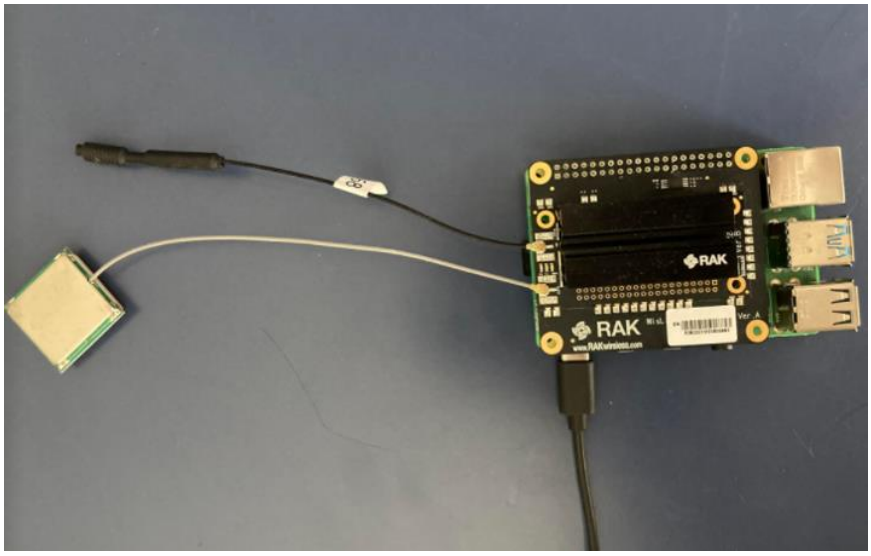


Figura 23. Imagen del montaje del gateway con módulo RAK 2245

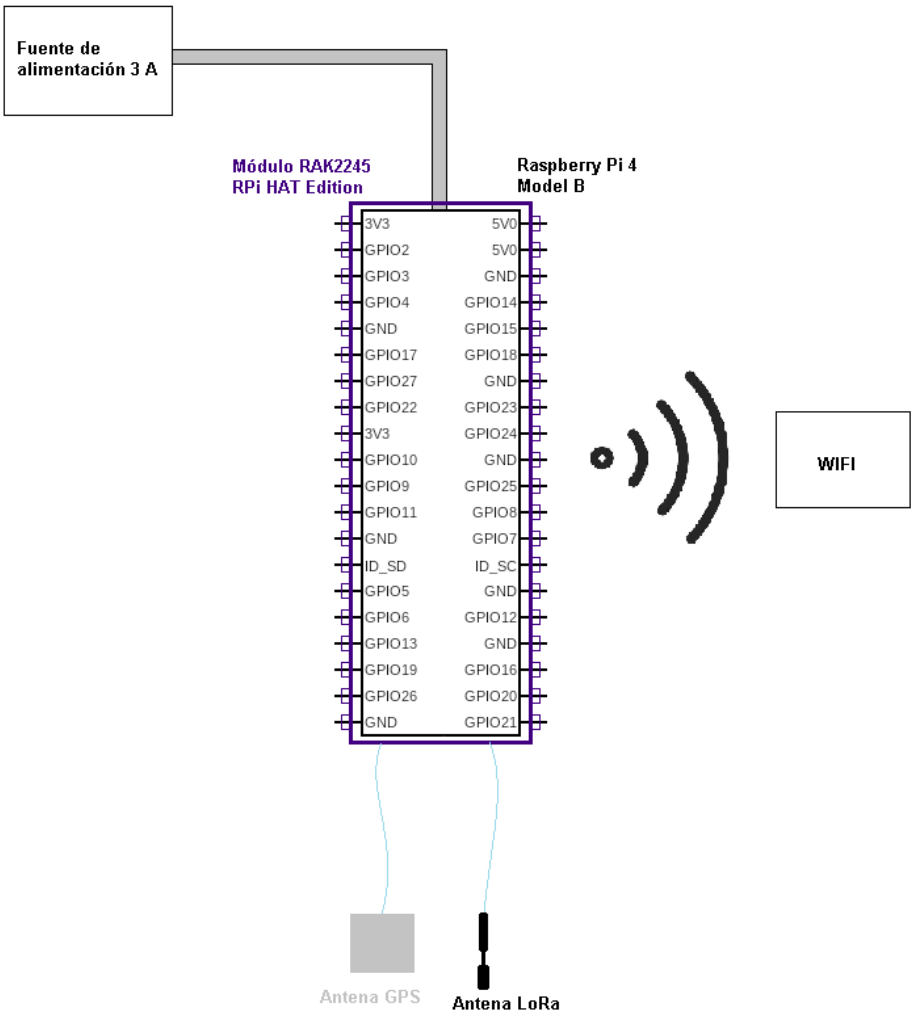


Figura 24. Esquemático del gateway con módulo RAK 2245

5.3 FUNCIONALIDADES

En esta sección se incluyen los códigos diseñados para los dos tipos de transmisiones realizadas, así como los resultados de estas transmisiones.

5.3.1 TRANSMISIÓN DE UN MENSAJE DE ALARMA POR ACTIVACIÓN DE TENG

5.3.1.1 Objetivo

Esta funcionalidad busca la transmisión de un mensaje de alarma cuando se pasa un umbral de tensión que es generado por el TENG. Se trata de una funcionalidad simple, pero gracias a las ventajas de autogeneración del TENG y de longitud y conectividad del protocolo LoRaWAN puede tratarse de un sistema de alarma revolucionario en un gran número de campos de aplicación.

Se trata de un sistema de alarma calibrable ya que simplemente hay que cambiar el valor de tensión umbral en el código. Esto permite que se pueda usar solo para detectar golpes fuertes en el TENG o más leves ya que la tensión pico del pulso está directamente relacionada con la fuerza aplicada. Contar con esta flexibilidad amplía el abanico de opciones ya que puede detectarse desde la caída de un bloque de metal hasta un ratón que pase por encima del TENG activándolo.

Una de las desventajas es que no se puede transmitir mensajes de alarma con una frecuencia mayor a 0,5 Hz debido a las normas de política justa de transmisión de mensajes de The Things Network. Otro problema que podría surgir es qué si hay un objeto que cae y activa la alarma, al quedarse el objeto ahí no se puede reactivar el sistema. Sin embargo, ambas problemáticas pueden solucionarse con distintas configuraciones de alarmas; se podrían poner diferentes capas con distintos sensores o diferentes etapas en las que distintos sensores se van activando aumentando la frecuencia de envío.

5.3.1.2 Programa

Para llevar a cabo esta funcionalidad se ha diseñado un programa en Arduino para la placa Arduino MKR WAN 1300 que digitaliza la señal de entrada por una de las entradas ADC del Arduino, tal y como se muestra en la Figura 12. El programa está compuesto por un archivo principal .ino (tipo Arduino) y un archivo .h (tipo *Header C++*).

En el archivo .h se almacenan el *appEUI* y el *appKEY* de la aplicación de The Things Network para que el Arduino MKR WAN sepa en qué aplicación está registrado cuando se conecte a la plataforma.

En el archivo principal del .ino se encuentra el código principal del programa. Primero, se inicializa el dispositivo y se conecta con The Things Network mediante el modo OTAA (*Over-The-Air-Activation*) accediendo a los valores almacenado en el .h. La conexión OTAA funciona de la siguiente manera:

1. El nodo solicita un join (o inicio de sesión) a la red con los datos de configuración y abre la ventana de recepción.
2. El Gateway recibe la solicitud y la envía al servidor.
3. El servidor verifica que el nodo este dado de alta y la llave de encriptación sea correcta.
4. Si es correcta asigna una sesión temporal y la envía por medio del gateway al nodo, si los datos son incorrectos rechaza el join.
5. El nodo recibe la sesión temporal y puede enviar datos a la red.

La principal ventaja de la conexión tipo OTAA es la seguridad ya que la sesión se dice: “se crea en el aire” y se renueva cada vez que el dispositivo pierde la conexión, es apagado o reiniciado, esto dificulta que alguien pueda robar la sesión y clonar el dispositivo (Sabas, 2021). Tras conectarse exitosamente, se establecen el ADR (*Adaptative Data Rate*) y el *Data Rate* de manera que ya se pueden empezar a enviar paquetes de datos a través del gateway.

Esta parte del código es imprescindible para cualquier inicialización de un módulo Arduino que vaya a establecer una comunicación LoRaWAN con The Things Network, por tanto, en la sección 5.3.2.2 se inicializará el programa de la misma manera.

La siguiente parte del código constituye el main() del programa (parte principal del programa). Se digitaliza el pulso utilizando la función `analogRead()`. Esta función digitaliza la señal de entrada devolviendo valores entre 0 y 1024, por tanto, la calibración de la alarma se realiza con un `if` que activa el envío del mensaje de alarma siempre que el valor de señal de entrada sea mayor que el un número entre 0 y 1023 elegido por el usuario. Si se supera el valor umbral, el código entra dentro del `if` y se llama a la función `modem.beginPacket()` para iniciar el envío del paquete y `modem.print("DANGER")` para fijar que es lo que se quiere enviar. Aquí se está enviando el paquete como texto ASCII tal cual, lo que puede suponer un aumento del tiempo de vuelo del mensaje, aunque como se comprueba en la sección 5.3.1.3, este tiempo es prácticamente inapreciable. Para terminar, se saca por el monitor serie si el envío ha sido exitoso o no gracias a la función `modem.endPacket(false)` que devuelve un 1 o un 0 respectivamente. Por último, hay un `delay(2000)` que equivale a 2 segundos de espera en el programa. Se pone para cumplir con el *Fair Policy Use* que requiere alrededor de 2 segundos de espera entre envío y envío para este tamaño de mensaje.

El código de este programa viene adjunto en el ANEXO I al final de este documento.

5.3.1.3 Resultados

Para simular los distintos estímulos que pueden provocar la activación del TENG se ha utilizado la mano golpeándolo con distintas fuerzas. El resultado ha sido exitoso y se ha cumplido con todas las expectativas. En la Figura 25 se muestra un ejemplo de cómo se recibe este mensaje en la página de The Things Network que puede monitorizarse desde cualquier parte del mundo con conexión a internet. La visualización se hace desde la sección *Data* dentro del menú de aplicación tal y cómo se ha explicado en la sección 26.

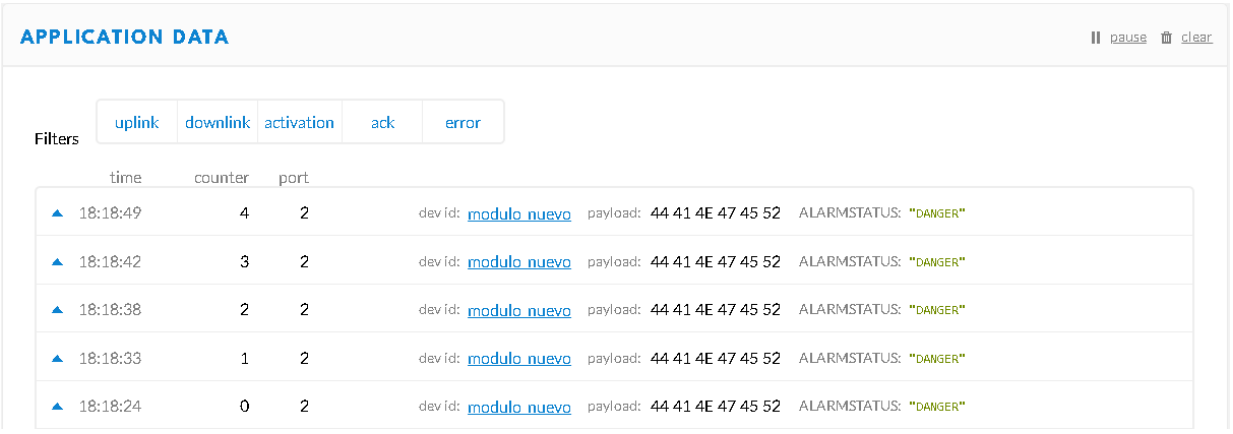


Figura 25. Prueba de transmisión de mensaje de alarma

Esta prueba se ha realizado golpeando el sensor con el dedo para simular cualquier objeto o mecanismo que pueda activarlo con una separación de 50 metros entre gateway y dispositivo emisor. Como se puede observar el mensaje llega sin problemas y siempre puede identificarse que dispositivo final está enviando el aviso como se ha explicado en la sección 5.1.2.4.

Si se extrae más información de alguno de los mensajes, como se ha hecho en la Figura 26, se puede comprobar que el tiempo de vuelo o *air time* estimado es de tan solo 36,096 ms (este tiempo puede variar con las condiciones del ambiente en el que se encuentren tanto nodo final como gateway y la distancia entre ellos). Esto significa que el receptor es informado del mensaje de alarma de manera casi instantánea.

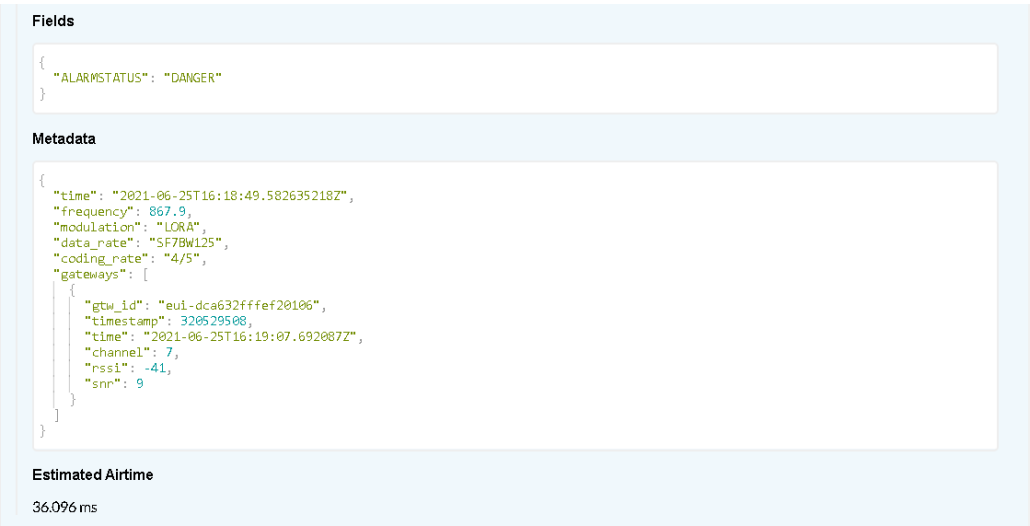


Figura 26. Análisis del tiempo de vuelo de un mensaje de alarma

Cabe destacar que esta funcionalidad funciona igual de bien, como es lógico, indistintamente del gateway que se utilice para transmitir el mensaje a TTN, es decir a internet.

5.3.2 TRANSMISIÓN DE VARIOS PULSOS DIGITALIZADOS

5.3.2.1 Objetivo

En esta segunda funcionalidad, se persigue la transmisión de uno o varios pulsos completos previamente digitalizados. Se quiere analizar si es posible conservar la frecuencia y la amplitud de la señal original en su llegada a TTN. De ser así se podría monitorizar vibraciones de todo tipo y detectar cuando se produce una alteración en las frecuencias de estas. Una aplicación sería la detección de vibraciones sísmicas o la respiración de una persona controlando la contracción del diafragma.

Ante este objetivo surgen dos problemas principales. El primero es que por los obligatorios intervalos entre distintos envíos no se puede monitorizar el pulso en tiempo real enviando los valores de este digitalizado. Por el contrario, es necesario grabar un fragmento de señal, digitalizarlo y enviar esta información para ser analizada cuando esta es recibida en The Things Network. El otro problema que surge es que, a pesar de tener un valor teórico de 209 bytes por paquete de datos enviado, el máximo que se permite en la práctica es de 64 bytes. Esto puede deberse a que la biblioteca de Arduino que se emplea para el envío <MKRWAN.h> limite el tamaño. No se ha encontrado información acerca del tamaño máximo que permite esta biblioteca, pero es la conclusión más razonable teniendo en cuenta que ya sucede en otras librerías como se ha explicado en la sección2.3. Por tanto, solo se pueden tomar 63 puntos por cada digitalización ya que el byte extra es para el tiempo tardado en tomar la muestra.

5.3.2.2 Programa

Para llevar a cabo esta funcionalidad se ha diseñado un programa en Arduino para la placa Arduino MKR WAN 1300 que lee la señal de entrada por una de las entradas ADC del Arduino, tal y como se muestra en la Figura 12. Al igual que el anterior el programa (explicado en la sección 5.3.1.2) está compuesto por dos archivos, uno .h y uno .ino. Ambos archivos son iguales y tienen la misma función con la excepción del `main()` del .ino. Además, también se produce una creación de variables antes de la configuración LoRa en este archivo.

Se comienza el código almacenando en la variable `inicio` el tiempo del microprocesador en ms en ese momento con el comando `millis()`. A continuación, con un bucle `for()` se almacenan 63 valores de la señal de entrada digitalizada con el comando `analogRead()`. Este comando tiene una frecuencia de muestreo teórica de 9600 Hz y práctica de 8600 Hz. Al ser esta demasiado elevada en algunas ocasiones y con solo 63 muestras por mensaje se puede incluir un `delay` entre muestreo y muestreo que reduce la frecuencia de muestreo y permite coger periodos de tiempo más largos. El bucle `for` se repite 63 veces almacenando 63 muestras de la señal digitalizada en un vector. Sin embargo, no se almacenan los valores que digitaliza `analogRead()` tal cual, sino que se escalan dividiéndose entre 5 para que el máximo valor posible, 1024, quepa en un byte, 256. Esto se hace porque si no se envían los datos como bytes se ocupa más tamaño en el mensaje y se reduce el número de puntos que se pueden enviar. Se está perdiendo un poco de resolución, pero si no se podrían enviar 63 puntos. Cuando el bucle termina se recoge en la variable `fin` el tiempo del microprocesador en ms en ese momento de nuevo con el comando `millis()`. Se calcula el tiempo transcurrido en el muestreo y se almacena en la última variable del vector donde se había almacenado la señal digitalizada. Para terminar, se emplea el mismo proceso de envío que en la sección 5.3.1.2 con la diferencia que se usa el comando `modemWrite()` en lugar de `modemPrint()` porque se está enviando la información como bytes y no como texto ASCII. Además, el `delay()` que se añade al final para cumplir con las especificaciones de *Fair Policy Use* pasa a ser de 6 segundos al tratarse de un mensaje con un tamaño más grande.

El código de este programa se adjunta, junto con el resto de los códigos, en el ANEXO I.

5.3.2.3 Resultados

En esta sección se incluyen las gráficas de varios pulsos antes de ser enviados y en su recepción The Things Network. Para cada ejemplo se incluye una imagen de los pulsos obtenida con el osciloscopio, una gráfica con la curva de los pulsos elaborada con Excel a partir de los datos extraídos del osciloscopio y la curva de los pulsos elaborada con Excel a partir de los valores recibidos en The Things Network (compensando la digitalización y el escalado del programa) y una tabla comparativa de las características más importantes a analizar. Todos los ensayos se han realizado con el mismo sensor fabricado con papel y polidimetilsiloxano PDMS.

Se han utilizado dos máquinas para generar los pulsos con unas frecuencias y fuerzas concretas (superiores a las que se podrían conseguir con la mano). Se ha variado la frecuencia y la fuerza para ver como respondían tanto el sensor TENG como el propio programa de envío de pulsos. Se han utilizado diferentes frecuencias de muestreo subiendo o bajando el *delay* del programa de Arduino en función de la frecuencia a la que se generasen los puntos para poder representarlos correctamente (se incluye el *delay* utilizado en cada caso). De ambas máquinas utilizadas para los experimentos se adjuntan las especificaciones técnicas en el ANEXO III.

Pulsos generados con máquina E3000 Linear-Torsion

La primera máquina utilizada se trata de la "*E3000 Linear-Torsion All-Electric Dynamic Test Instrument*" de la marca INSTRON y se encuentra en el Laboratorio de Materiales de ICAI. Utiliza una tecnología electromagnética y se controla mediante ordenador. En esta máquina los valores que se varían son la frecuencia de pulsación y la distancia a la que se separan las placas que generan las pulsaciones. Se adjunta una imagen de la máquina utilizada en la Figura 27.

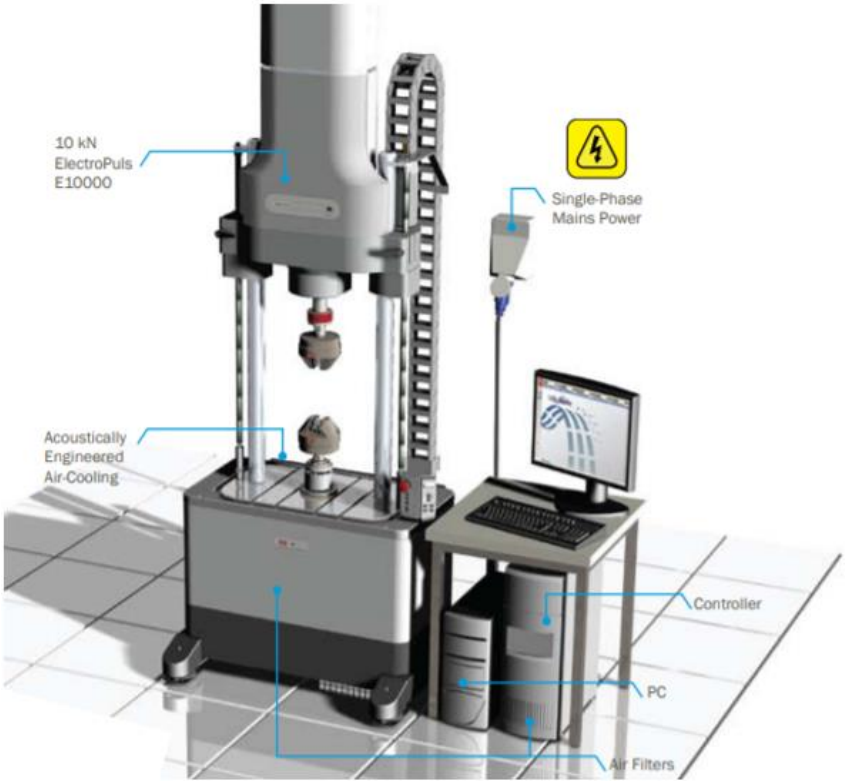


Figura 27. Imagen de la máquina E3000 Linear-Torsion All-Electric Dynamic Test Instrument

En la Figura 27 se observan las dos mordazas con las que cuenta la máquina para realizar distintos ensayos. En este caso, se colocó en la mordaza inferior un bloque de goma (*silicon block*) para amortiguar el impacto. Sobre la base de este bloque amortiguador se fijó el TENG con un adhesivo y se apretó con la mordaza superior. Una vez el montaje estaba realizado se iniciaron las vibraciones con elongaciones determinadas. El sensor comienza apretado a una fuerza determinada (aproximadamente 180 N) para luego empezar a funcionar en fatiga (extensión y compresión) y producir vibraciones de diferentes frecuencias. Los ciclos de elongación funcionan siguiendo el proceso de la Figura 28 donde la distancia entre las mordazas que se ha fijado en el ordenador es la diferencia entre la línea azul y roja. La curva de la izquierda representa un ciclo de vibración donde el TENG comienza con una fuerza de 300 N al inicio del ciclo y se relaja hasta los 40 N.

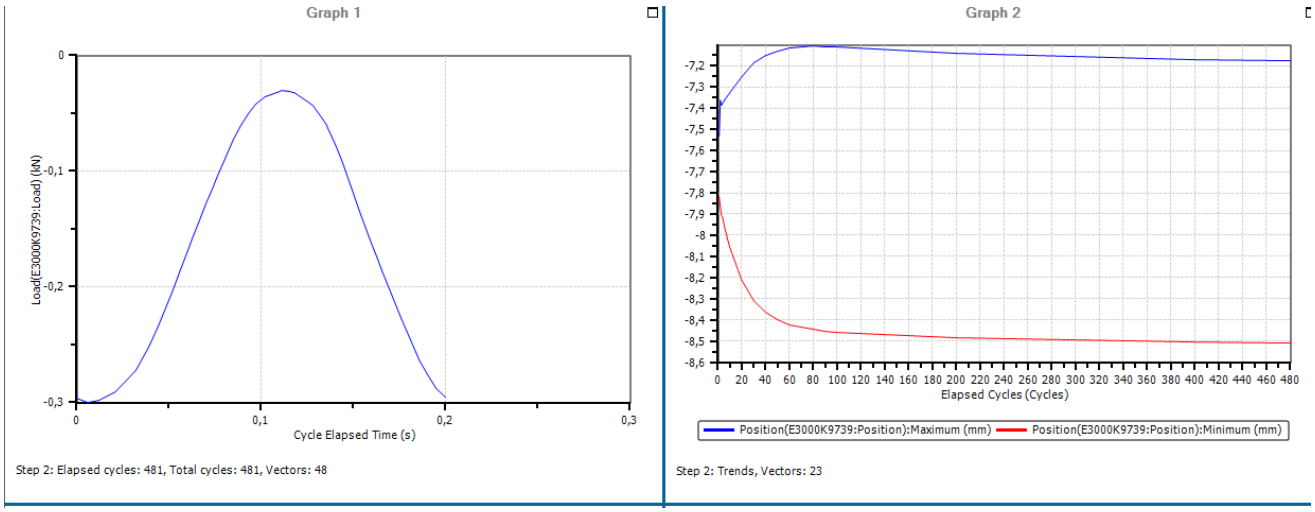


Figura 28. Ciclo de trabajo de la máquina E3000 Linear-Torsion All-Electric Dynamic Test Instrument

Las medidas del osciloscopio y de The Things Network se obtuvieron por separado, pero con la máquina funcionando bajo los mismos parámetros de frecuencia y distancia por lo que en esencia se trata del mismo pulso.

- **Pulso 1. PAPER+PDMS. Frecuencia: 5 Hz Desplazamiento: 1 mm**

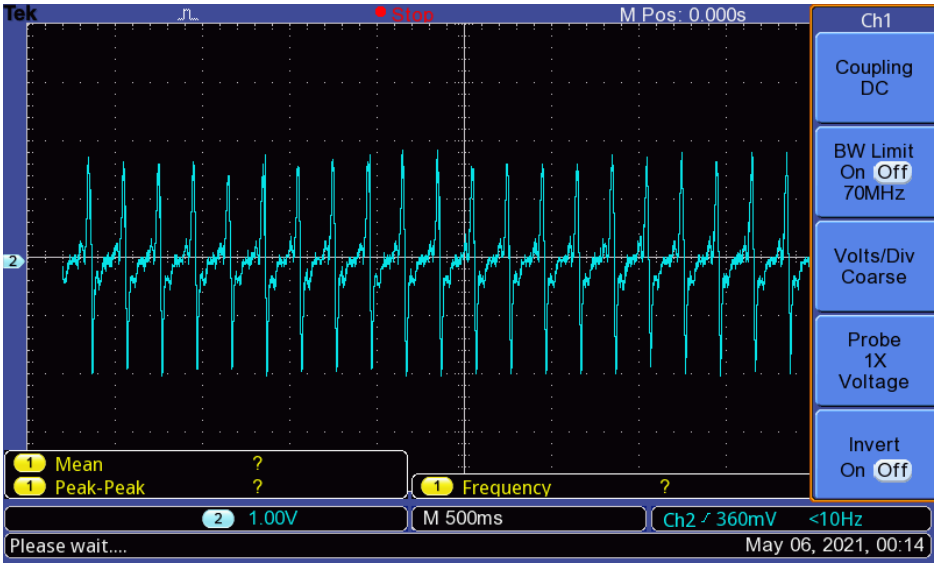


Figura 29. Imagen del osciloscopio de pulsos generados por TENG con máquina E3000 Linear-Torsion. Frecuencia: 5 Hz Desplazamiento: 1 mm

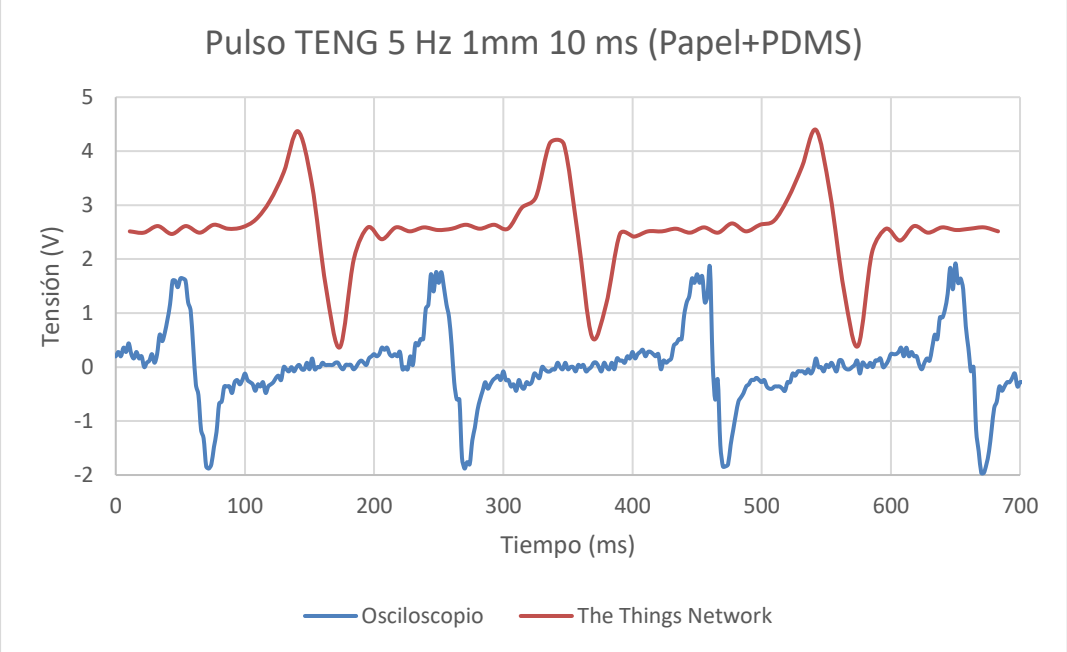


Figura 30. Gráfica comparativa de pulsos generados por TENG con máquina E3000 Linear-Torsion. Frecuencia: 5 Hz Desplazamiento: 1 mm Delay: 10 ms

Como se puede observar en la Figura 30 se han obtenido pulsos muy similares del osciloscopio y con la representación obtenida a través de los datos de The Things Network. La gráfica que se ha obtenido con la información transmitida por el protocolo LoRaWAN es prácticamente igual que la del osciloscopio, pero desfasada en tensión debido a un ruido medio que aparece en la conversión analógica-digital del Arduino MKR WAN 1300. También se observa que el ruido del osciloscopio es mayor que en el MKR WAN 1300 porque hay mayor frecuencia de muestreo en el osciloscopio y por tanto más puntos. La baja frecuencia de muestreo del MKR WAN 1300 y el propio programa de Excel que une los puntos tomados con curvas suaves actúan como un filtro paso alto eliminando el ruido blanco de la señal. En la siguiente Tabla 3 se ha realizado una comparación de los cuatro parámetros más importantes que se analizarán en el Capítulo 6.

Parámetro	**Osciloscopio**	**The Things Network**
Frecuencia [Hz]	5	5,12
Amplitud (Máximo-Mínimo) [V]	3,88	4
Valor Medio [V]	0	2,5
Distorsión formal	-	No

Tabla 3. Tabla de comparación de parámetros de Pulso 1

- **Pulso 2. PAPER+PDMS. Frecuencia: 40 Hz Desplazamiento: 0,3 mm**

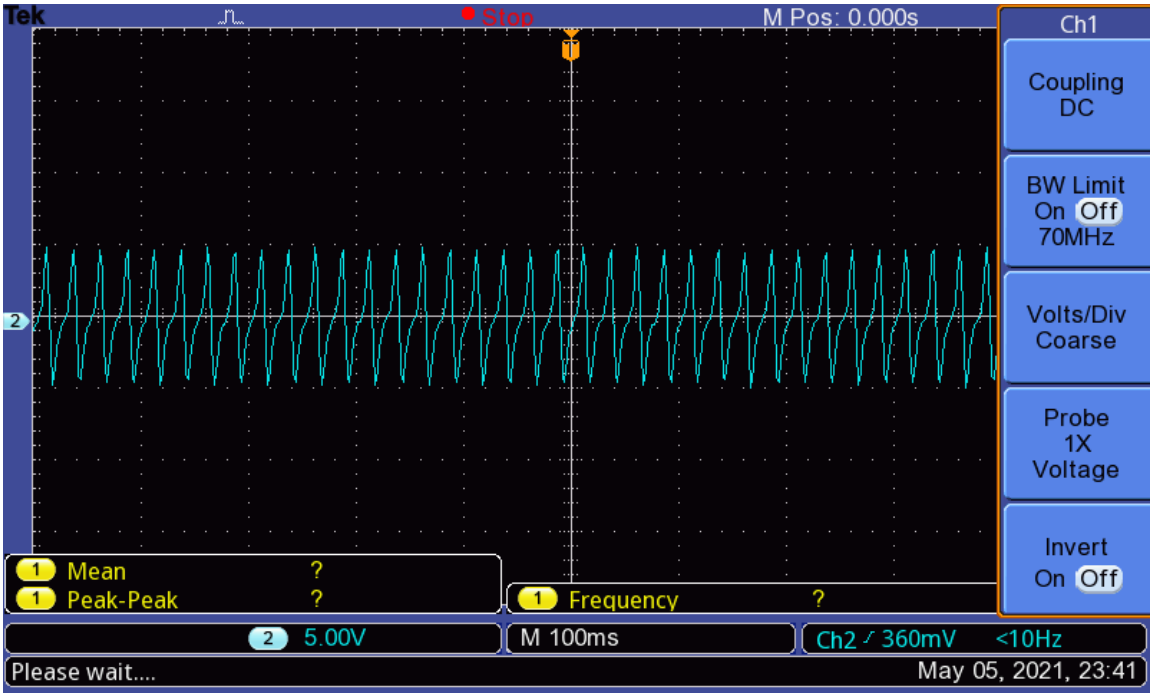


Figura 31. Imagen del osciloscopio de pulsos generados por TENG con máquina de prueba universal. Frecuencia: 40 Hz Desplazamiento: 0,3 mm

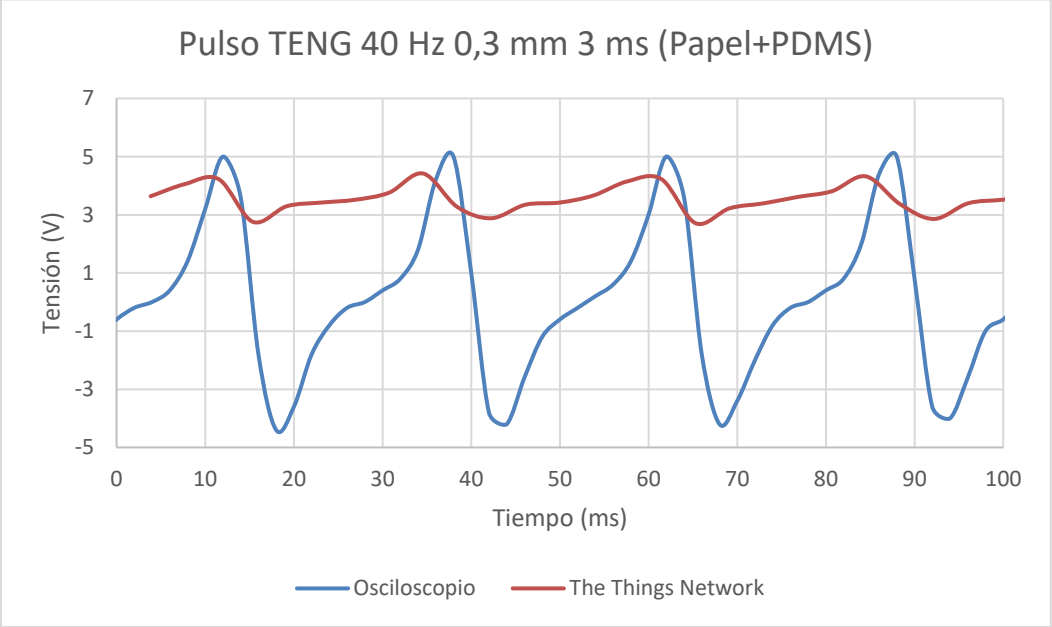


Figura 32. Gráfica comparativa de pulsos generados por TENG con máquina E3000 Linear-Torsion. Frecuencia: 40 Hz Desplazamiento: 0,3 mm Delay: 3 ms

En este primer envío de un pulso de 40 Hz se ha probado con un delay de 3 ms y los resultados han sido muy malos como se ve claramente en la Figura 32. Al aumentar mucho la frecuencia del pulso, es necesario aumentar mucho también la frecuencia de muestreo para que el programa sea capaz de detectarlo y lo pueda representar con exactitud. En este caso no se ha conseguido obtener una frecuencia de muestreo adecuada por lo que el pulso, aunque conserva la frecuencia del original, no mantiene la amplitud de este, sin deberse a una saturación. El programa toma muestras en la subida y en la bajada e intuye un pico, pero no llega a representar la magnitud de este con fidelidad. Se ha añadido esta gráfica para ejemplificar los peligros de no utilizar un delay adecuado, y por tanto una frecuencia de muestreo adecuada. De nuevo se puede volver a apreciar un desfase de tensión en el valor medio de los pulsos causado por el MKR WAN 1300. Como en el caso anterior, se ha añadido la Tabla 4 donde se realiza la comparación de los cuatro parámetros más relevantes para este proyecto.

Parámetro	**Osciloscopio**	**The Things Network**
Frecuencia [Hz]	41,6	43,3
Amplitud (Máximo-Mínimo) [V]	9,4	1,68
Valor Medio [V]	0	3,5
Distorsión formal	-	Sí

Tabla 4. Tabla de comparación de parámetros de Pulso 2.A

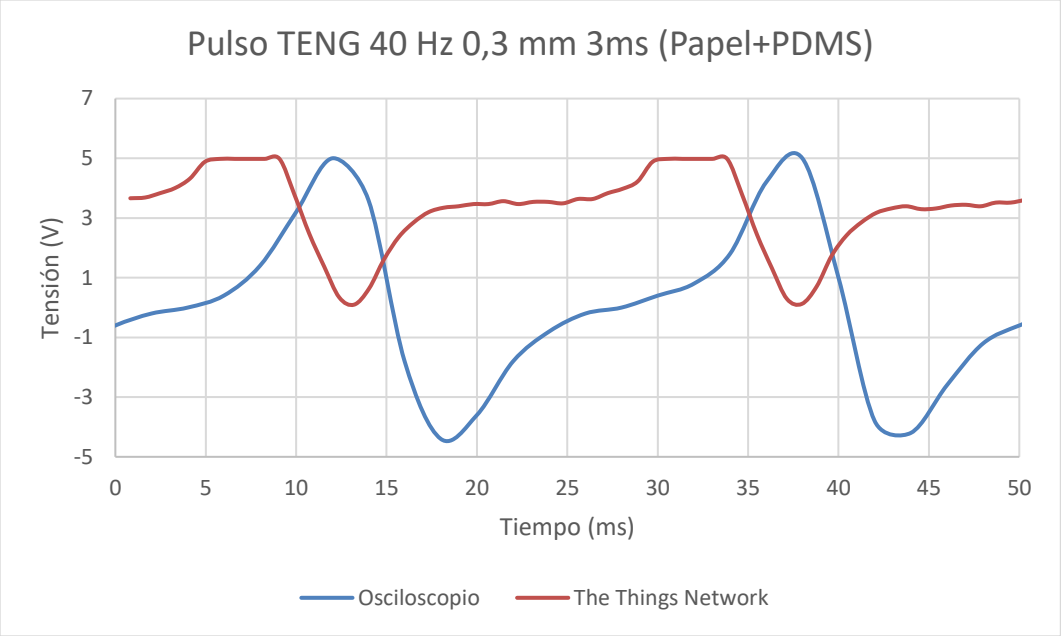


Figura 33. Gráfica comparativa de pulsos generados por TENG con máquina E3000 Linear-Torsion. Frecuencia: 40 Hz Desplazamiento: 0,3 mm Delay: 0 ms

Para el pulso representado en la Figura 33 se ha empleado un delay de 0 ms de manera que se consiga la máxima frecuencia de muestreo posible. En este caso el pulso si se parece al original ya que se está muestreando mucho más rápido. Sin embargo, al tratarse de un pulso con unos valores de pico por encima de los 2,5 V, y teniendo en cuenta el ya comentado ruido añadido por el propio dispositivo, el Arduino MKR WAN satura sin la posibilidad de muestrear puntos limitándolos a 5 V como se puede observar en la Figura 33. Este es un claro ejemplo de dos situaciones que se discutirán con más detalle en el Capítulo 6. la saturación causada por los límites del Arduino MKR WAN y el dilema entre número de pulsos representados y calidad de las representaciones.

Parámetro	**Osciloscopio**	**The Things Network**
Frecuencia [Hz]	41,6	40,38
Amplitud (Máximo-Mínimo) [V]	9,4	5
Valor Medio [V]	0	3,5
Distorsión formal	-	Sí (Saturación)

Tabla 5. Tabla de comparación de parámetros de Pulso 2.B

La segunda máquina se trata un equipo de Análisis Mecánico-Dinámico o *Dynamic Mechanical Analyzer*, en concreto el "DMA Q800" de la marca TA Instruments y se encuentra en IMDEA Materiales. En esta máquina se controlan la frecuencia de pulsación y la fuerza de manera directa. En la Figura 34 se ilustra el montaje realizado en el equipo de DMA para el envío de los pulsos. Al igual que en el caso anterior se fijó el TENG a la plataforma del DMA con un adhesivo. Se siguió el mismo procedimiento de obtención de medidas que en el caso anterior, de nuevo manteniendo fijos los parámetros de frecuencia y fuerza en este caso.

Figura 34. Montaje del experimento de envío con el equipo DMA

Pulsos generados con equipo de Análisis Mecánico-Dinámico (DMA)

- **Pulso 3. PAPER+PDMS. Frecuencia: 10 Hz Fuerza: 3 N**

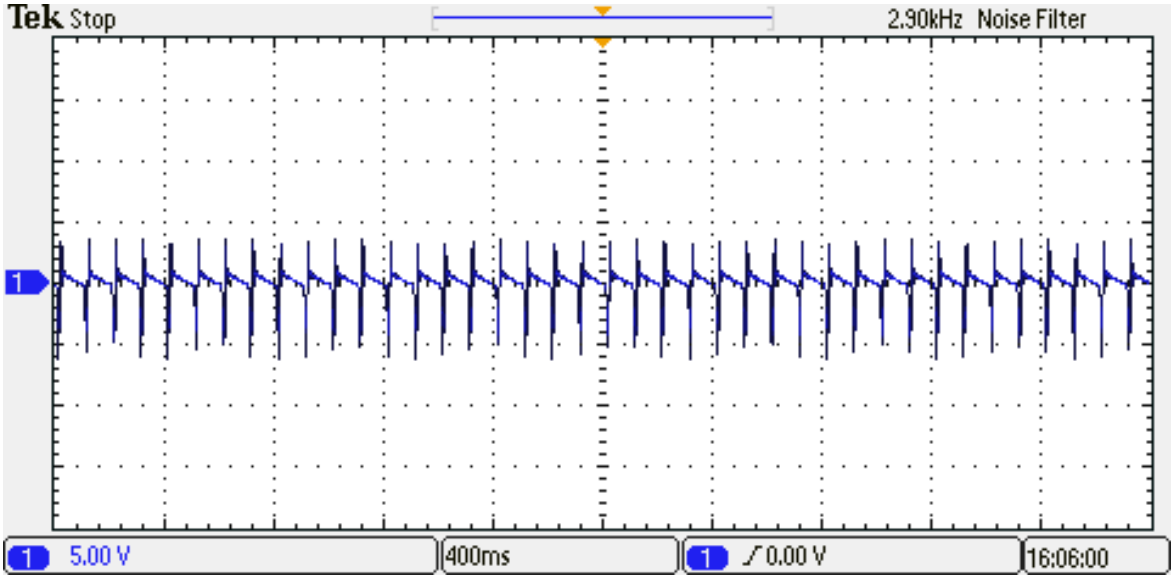


Figura 35. Imagen de osciloscopio de pulsos generados por TENG con DMA.

Frecuencia: 10 Hz Fuerza: 3 N

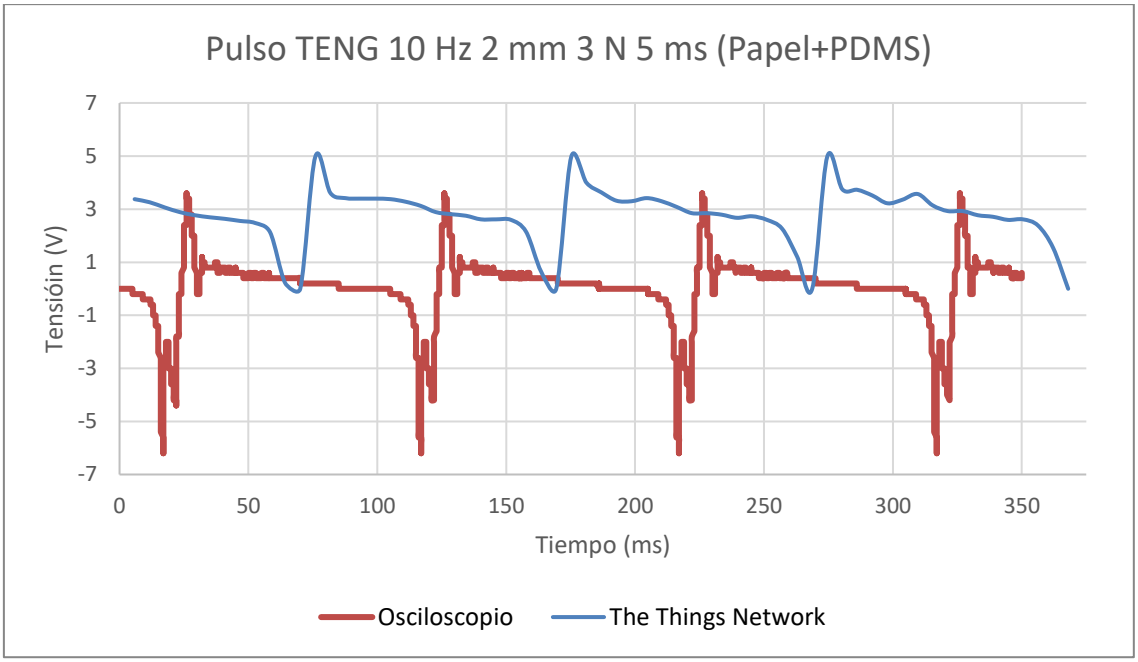


Figura 36. Gráfica comparativa de pulsos generados por TENG con equipo DMA.

Frecuencia: 10 Hz Fuerza: 3 N Distancia: 2 mm Delay: 5 ms

En la Figura 36 podemos observar que los pulsos obtenidos con el osciloscopio y con The Things Network son muy similares entre sí. Como siempre, aparece el desfase en tensión generado por el ruido que genera que el valor medio de la señal de The Things Network no sea 0. Es interesante como en la gráfica de TTN no se alcanzan los mismos valores de tensión de pico que con el osciloscopio, sin embargo, no parece haber saturación. Lo que ocurre es que sí que hay saturación, pero al tener una subida y bajada tan rápida y no estar utilizando la máxima frecuencia de muestreo no se toma más de una muestra en el punto máximo. Sin embargo, si no existiese limitación de tensión en el Arduino la muestra que se tomaría en el pico no sería de 5 V sino de 6 V. Si aumentásemos la frecuencia de muestreo sí se observaría la saturación como se ha podido comprobar en la Figura 33. Se ha elaborado la Tabla 6 para recoger las comparaciones de los parámetros que se pretenden estudiar en el análisis de los resultados.

Parámetro	**Osciloscopio**	**The Things Network**
Frecuencia [Hz]	10	10,07
Amplitud (Máximo-Mínimo) [V]	9,8	5
Valor Medio [V]	0	3
Distorsión formal	-	No

Tabla 6. Tabla de comparación de parámetros de Pulso 3

- **Pulso 4. PAPER+PDMS. Frecuencia: 10 Hz Fuerza: 3 N**

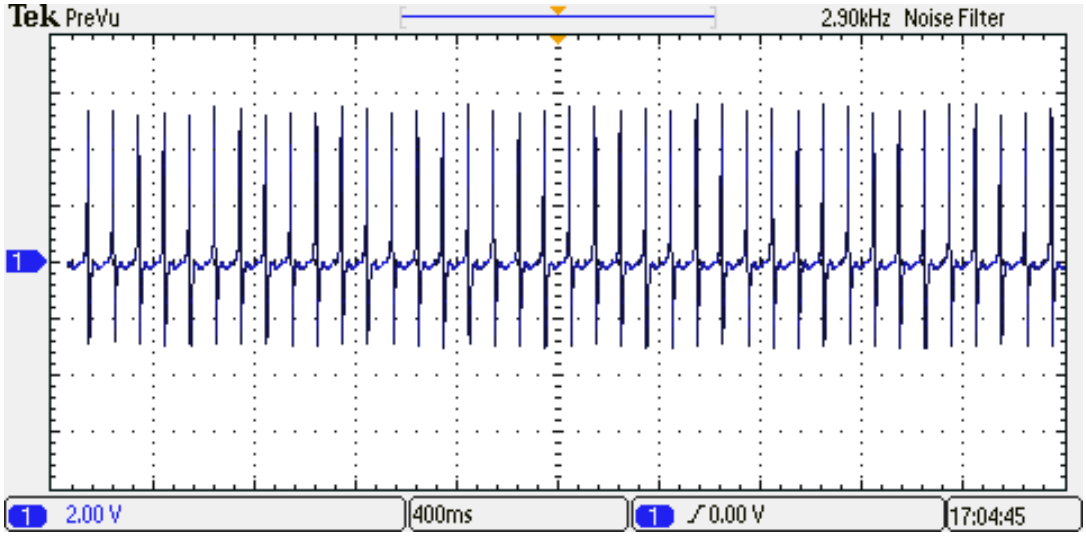


Figura 37. Imagen de osciloscopio de pulsos generados por TENG con DMA.

Frecuencia: 10 Hz Fuerza: 3 N

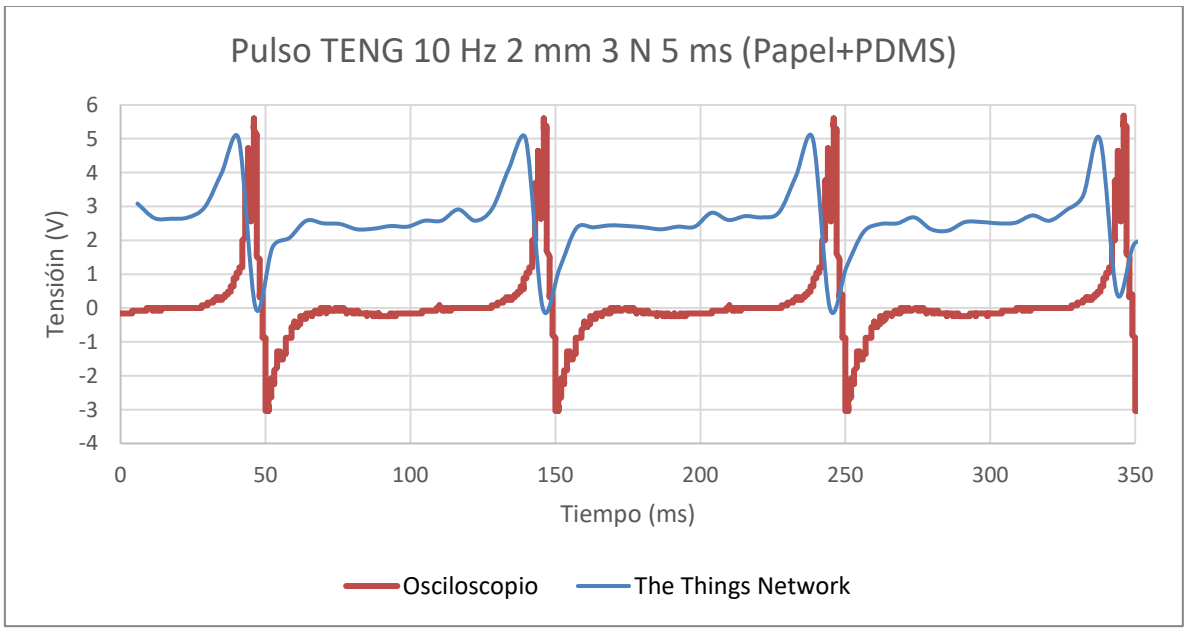


Figura 38. Gráfica comparativa de pulsos generados por TENG con equipo DMA.

Frecuencia: 10 Hz Fuerza: 3 N Distancia: 2 mm Delay: 5 ms

En la Figura 38 se han empleado los mismos ajustes que en la Figura 36 sin embargo se ha dado la vuelta al TENG para observar como el pulso cambia de forma, experimentado primero la subida y luego la bajada al contrario que en el caso anterior. En líneas generales sucede lo mismo que en el pulso anterior, con un valor medio provocado por el ruido, limitación de tensión provocada por el MKR WAN 1300 y se mantiene la frecuencia. La Tabla 7 recoge la información de frecuencia, amplitud y valor medio comparando la gráfica obtenida con el osciloscopio y la obtenida a partir de los valores de The Things Network.

Parámetro	**Osciloscopio**	**The Things Network**
Frecuencia [Hz]	10	10,09
Amplitud (Máximo-Mínimo) [V]	8,64	5
Valor Medio [V]	0	2,5
Distorsión formal	-	No

Tabla 7. Tabla de comparación de parámetros de Pulso 4

- **Pulso 5. PAPER+PDMS. Frecuencia: 10 Hz Fuerza: 5 N**

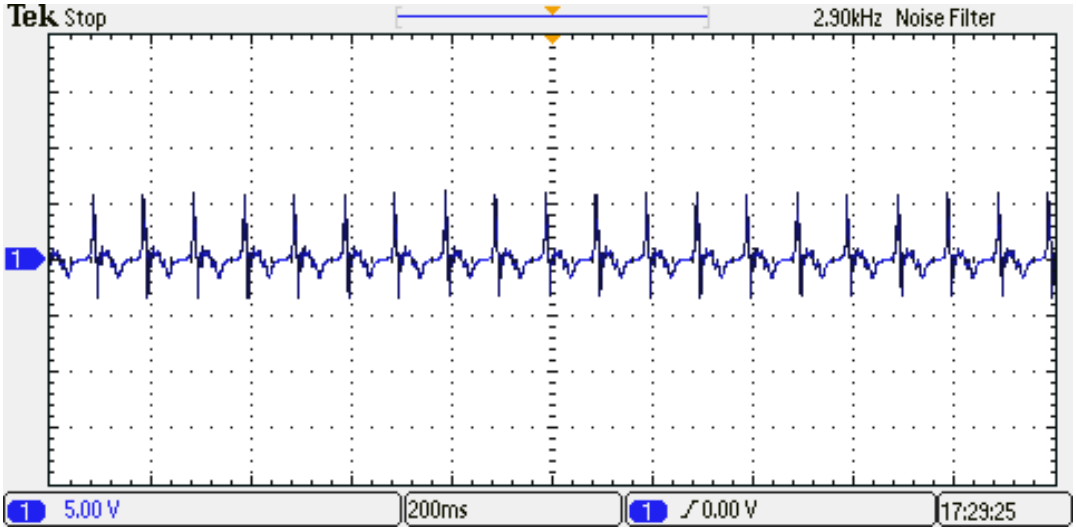


Figura 39. Gráfica de The Things Network de pulsos generados por TENG con DMA.

Frecuencia: 10 Hz Fuerza: 5 N

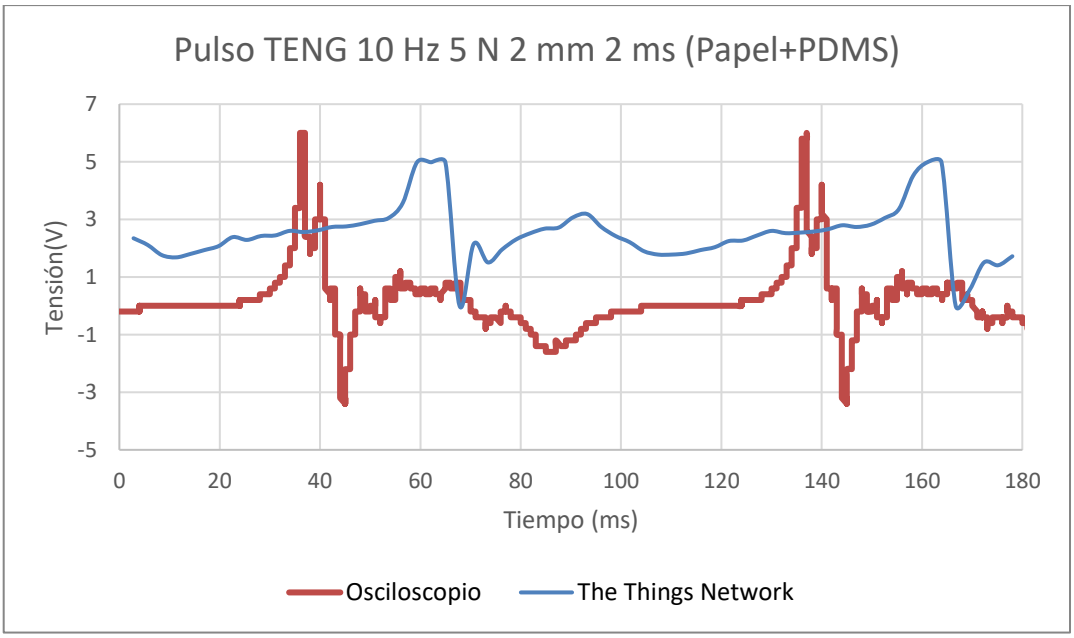


Figura 40. Gráfica comparativa de pulsos generados por TENG con equipo DMA.

Frecuencia: 10 Hz Fuerza: 5 N Distancia: 2 mm Delay: 2 ms

En este envío se ha configurado el equipo DMA para que genere una fuerza de 5 N. Se puede comprobar comparando la Figura 40 frente a la Figura 38 que el TENG genera mayores picos de voltaje si se aplica más fuerza (9,4 V de amplitud pico- pico para 5 N y 8,6 V para 3 N). El aumento de fuerza también distorsiona un poco la forma del pulso teniendo una etapa de recuperación más larga que en los casos anteriores. De nuevo se vuelve a observar saturación debido al elevado pico de tensión, pero en líneas generales el pulso está representado con cierta precisión manteniendo como siempre la frecuencia intacta. En la Tabla 8 se realiza la última comparativa que servirá para un análisis más detallado en el Capítulo 6. de los parámetros comparados.

Parámetro	**Osciloscopio**	**The Things Network**
Frecuencia [Hz]	10	10,11
Amplitud (Máximo-Mínimo) [V]	9,4	5
Valor Medio [V]	0	2,5
Distorsión formal	-	Sí (Saturación)

Tabla 8. Tabla de comparación de parámetros de Pulso 5

Capítulo 6. ANÁLISIS DE RESULTADOS

En este capítulo se va a realizar un análisis de los resultados obtenidos en la sección 5.3.2. Se han incluido un gran número de gráficas con el objetivo de poder analizar si existe distorsión en el pulso recibido, que factores influyen en esta distorsión y cómo solucionar este problema si es que existe.

6.1 VALOR MEDIO

Una de las diferencias más notables entre la señal original y la señal recibida, es que en la primera nuestro valor medio siempre es 0 y el pulso está formado por dos picos de tensión, uno negativo y otro positivo (el orden depende de la polaridad con la que se realicen las medidas, pero es irrelevante en el análisis que se pretende realizar en este capítulo).

Este valor medio, aparece como ruido blanco inherente al Arduino MKR WAN 1300. Por un lado, tiene el aspecto positivo que permite visualizar los picos negativos y positivos (aunque desfasados en tensión) ya que, de lo contrario, como las placas de Arduino no son capaces de detectar tensiones negativas, en el pico negativo se vería una señal continua en 0. Sin embargo, no todo son ventajas por el hecho de que este valor medio no es constante en todos los envíos. Como se puede apreciar en el Pulso 1, Pulso 4 y Pulso 5 existe un valor medio de 2,5 V, mientras que en el Pulso 2 y Pulso 3 el valor medio es de 3,5 V y 3 V respectivamente. No existe ningún parámetro del pulso original del que depende este ruido base. En desarrollos futuros que se representase el pulso de manera automática, esto podría suponer complicaciones.

Idealmente, se debería eliminar este ruido consiguiendo que la señal base fuese 0 y añadir en paralelo una tensión constante de 2,5 V que permita visualizar picos positivos de 2,5 V y negativos de -2,5 V. Aunque puede parecer que es eliminar una cosa para poner la misma, la diferencia reside en que esta tensión sí estaría controlada y podría corregirse en su posterior representación de manera automática.

6.2 AMPLITUD

Otro de los problemas que aparece al enviar pulsos es que nunca se consigue visualizar una amplitud superior a 5 V. Esto se debe a que este es el rango de lectura del Arduino. Esto supone un problema ya que a excepción del Pulso 1 los valores de amplitud no cuadran en absoluto con la señal original como se puede comprobar al superponer las curvas en la misma gráfica.

Si se ignoran los picos negativos y se consigue un valor medio de 0 V el pico máximo que va a poder detectar nuestro programa será de 5 V. Por otro lado, si se persigue una mayor fidelidad en la representación del punto completo y se fija el valor medio en 2,5 V solo se podrán monitorizar picos positivos de 2,5 V y negativos de -2,5 V.

El sistema puede seguir funcionando perfectamente para pulsos con menores amplitudes de pico. Por tanto, una solución es diseñar los TENG de manera que se reduzca su capacidad para generar voltaje. En caso de que esto sea muy costoso o resulte imposible existe la posibilidad de poner un divisor resistivo de tensiones que reduzca la amplitud del pulso. Es importante que se utilicen resistencias de valores bajos dado que el TENG genera muy poca intensidad y si se diseña un divisor con una resistencia de salida muy grande, toda la corriente sería absorbida por esta resistencia y se perdería la señal. Esta reducción debería corregirse una vez se muestree la señal con los valores de The Things Network. Tiene el aspecto negativo de que se pierde resolución. Como última alternativa, se puede plantear un cambio de dispositivo final, a uno que maneje márgenes de tensión superiores al Arduino MKR WAN 1300.

6.3 FRECUENCIA

La frecuencia es el parámetro que mejor se conserva ya que mantiene los valores originales en todos los pulsos enviados como se ilustra en Pulso 1, Pulso 2, Pulso 3, Pulso 4 y Pulso 5. El cálculo se ha realiza como el inverso del tiempo que pasa entre dos picos de tensión (ya sean positivos o negativos).

Dado que en este apartado no ha habido ningún problema que solucionar solamente se proponen una forma de mejorar la detección de frecuencia y pasa por aumentar la frecuencia de muestreo como se explicará a continuación en la sección

6.4 DISTORSIÓN FORMAL

La distorsión formal hace referencia cuanto y como se distorsiona la forma de la señal desde que es generada por el TENG hasta que es muestreada con los valores de The Things Network.

Realmente no aparecen distorsiones formales serias en las que se obtenga una señal completamente distinta a la original a excepción de una de las representaciones. Esto se debe a que en todo momento se está cumpliendo con el Teorema Fundamental del Muestreo que establece que la frecuencia de muestreo tiene que ser al menos 2 veces superior a la frecuencia de la señal que se quiere muestrear. Cuanto mayor sea la frecuencia de muestreo, se toman más muestras en menos tiempo y se puede realizar una representación más fiel de la realidad.

Lo ideal por tanto sería utilizar siempre la menor frecuencia de muestreo, sin embargo, como se ha comentado en la sección 5.3.2.2 al existir un límite de bytes que se pueden enviar, hay que encontrar un compromiso entre poder representar la señal con una frecuencia de muestreo adecuada y representar más de un pulso completo (ya que de otra manera no se podría obtener la frecuencia de pulsos). Esto se ilustra perfectamente en la comparación de la Figura 32 y Figura 33. Se ha muestreado la misma señal, pero con diferentes *delays*, es decir, diferentes frecuencias de muestreo. En la Figura 32 se ha utilizado una frecuencia de muestreo inferior por lo que se pierde información de la señal, aunque se sigue cumpliendo el Teorema Fundamental del Muestreo (261,41 Hz > 40 Hz), a costa de representar hasta 10 picos. En la Figura 33 se puede identificar con mucho más detalle el pulso generado, pero solo se llega a muestrear 2 pulsos completos.

Se ha elaborado la siguiente Tabla 9 que establece las frecuencias de muestreo en función del *delay* usado en el código de Arduino. La columna de tiempo de recogida se ha obtenido de manera experimental, el periodo de muestreo se calcula como el tiempo de recogida entre el número de muestras tomadas (siempre 63 en nuestro caso) y la frecuencia de muestreo se calcula como el inverso del periodo de muestreo. Como se puede comprobar, al haberse trabajado con frecuencias de señal de máximo 40 Hz, en ningún momento se ha incumplido el Teorema Fundamental del Muestreo.

Delay (ms)	**Tiempo de recogida (ms)**	**Periodo de muestreo (ms)**	**Frecuencia de muestreo (Hz)**
0	52	0.825396825	1211.538462
1	115	1.825396825	547.826087
2	178	2.825396825	353.9325843
3	241	3.825396825	261.4107884
4	304	4.825396825	207.2368421
5	367	5.825396825	171.6621253
6	430	6.825396825	146.5116279
7	493	7.825396825	127.7890467
8	556	8.825396825	113.3093525
9	619	9.825396825	101.7770598
10	682	10.82539683	92.37536657

Tabla 9. Frecuencias de muestreo para distintos delays

Capítulo 7. CONCLUSIONES Y TRABAJOS FUTUROS

En esta sección se va a establecer si se ha logrado la consecución o no de los objetivos fijados mientras que se extraen conclusiones sobre el desarrollo de estos. También se añadirán recomendaciones para seguir investigando en la dirección que marca este proyecto.

7.1 OBJETIVOS CUMPLIDOS

El primer objetivo fijado en la sección 4.2 era la fabricación de un sistema de comunicación IoT. Para lograr este objetivo, se seleccionó la tecnología LoRa y el protocolo de comunicación LoRaWAN. Se ha logrado obtener un sistema funcional que permite el envío de información y que ofrece tanto una gran cantidad de beneficios como algunas dificultades para uno de los objetivos propuestos.

El segundo objetivo marcado en la sección 4.2 era diseñar un programa que mandase un mensaje de alarma a través del sistema de comunicación IoT cuando se activase el TENG. Este objetivo se ha logrado con éxito. El programa diseñado permite elegir el mensaje que se quiere enviar, se puede calibrar la tensión a la que se desea que se mande este mensaje (de manera que no se active el sensor) y gracias a la encriptación del protocolo LoRaWAN el envío siempre goza de una alta seguridad. Su recepción puede monitorizarse con The Things Network desde cualquier parte del mundo de manera que se distingan los diferentes dispositivos emisores del mensaje de alarma gracias al identificador de dispositivo *device_ID* que se fija en el registro de este en la plataforma.

El tercer y último objetivo fijado consistía en el envío de varios pulsos completos y el análisis en la recepción de estos. Este objetivo se ha llevado a cabo, aunque los resultados no han sido tan positivos como se esperaba. La tecnología montada junto con el programa diseñado es capaz de realizar la transmisión de información, pero ésta se recibe distorsionada a no ser que se mantengan unos límites muy concretos y se haya establecido

previamente un *delay* adecuado para la frecuencia a emplear. En la sección siguiente 7.2 se va a realizar un breve análisis sobre los aspectos positivos de la tecnología LoRa y el protocolo LoRaWAN para esta funcionalidad.

7.2 TECNOLOGÍA LORA Y PROTOCOLO LORAWAN PARA ENVÍO DE PULSOS COMPLETOS

Se ha establecido una lista de aspectos positivos y cómo influyen en la consecución de este objetivo:

- Capacidad para transmitir información de manera inalámbrica en distancias mayúsculas. Como se ha comentado al inicio de esta memoria, en la sección 2.2, las distancias que puede alcanzar el protocolo LoraWAN pueden superar los 500 km en situaciones extremadamente favorables y los 10 km en interiores con condiciones normales urbanas (Allan, 2019). Esto ayuda a que los dispositivos no tengan que estar muy cerca en el momento del envío y se puedan monitorizar pulsos generados por en personas o vehículos en movimiento, siempre que no se alejen a más de esta distancia sin entrar en el radio de otro gateway.
- Los dispositivos finales presentan un bajo consumo de potencia. La placa Arduino MKR WAN 1300 se alimenta con 2 pilas AAA siguiendo la hoja de características, pero realmente la mejor forma de alimentarla es con una batería Li-Po (polímero de iones de litio) de 3,7 V. Para cualquier funcionalidad a gran escala, este bajo consumo reduciría los costes de manera notable frente a otras tecnologías como el Wifi o Bluetooth.
- La información transmitida no sufre ningún tipo de distorsión en la transmisión. Los problemas que han surgido en este trabajo han estado relacionados con la digitalización del pulso. Sin embargo, la información que una vez digitalizada es enviada, llega intacta a su destino y sin ningún tipo de distorsión.
- Gracias a The Things Network se pueden realizar proyectos de investigación a escalas pequeñas sin la necesidad de crear tu propia red de transmisión de datos y ofrece algunas integraciones para el almacenamiento de datos; simples y de costo

gratuito y más completas, pero de suscripción económica. Para la investigación individual la plataforma de The Things Network facilita el proceso lo que permite que trabajos como este sean posibles sin un conocimiento muy avanzado de tecnologías de la comunicación.

Por otro lado, hay aspectos generales del protocolo y la tecnología y algunos concretos de la placa elegida que presentan dificultades para conseguir una transmisión fiel de los pulsos.

- El protocolo LoRaWAN no está diseñado para el envío de mucha información ni de manera muy frecuente. Además, a placa escogida, por el hecho de utilizar la biblioteca <Arduino.h>, reduce (no de manera drástica) aún más el tamaño máximo de los mensajes. La limitación de frecuencia de envío imposibilita la representación del pulso en tiempo real y la limitación de tamaño obliga a tener que comprometer frecuencia de muestreo y tiempo total de muestreo tal y como se ha explicado en la sección 6.4.
- En concreto la placa MKR WAN 1300 ha resultado tener mucho ruido por sus entradas analógicas. Se ha trabajado con 5 placas y en todas existe una señal de ruido de continua inherente a la placa. Es cierto que el ruido blanco no es muy elevado, pero a medida que se usa la placa, este va aumentando. Aunque en algunas ocasiones pueda llegar a resultar incluso útil (como se explica en la sección 6.1) se trata de un elemento que no se controla y que complica la posterior representación del pulso.
- Los límites de tensión de la placa Arduino MKR WAN 1300 son muy pequeños en comparación con las tensiones que puede llegar a generar un TENG. Surgen muchos problemas de saturación en 0 V y 5 V de manera que impiden ver una versión real del pulso.

En resumen, se trata de una tecnología con mucho potencial y aún mucho margen de mejora. Actualmente se está centrando en evolucionar hacia otro tipo de aplicaciones. Sin embargo, en envíos de mensajes de alarma cumple todos los requisitos necesarios y proporciona resultados muy positivos y para el envío de pulsos habría que investigar con

dispositivos finales distintos al Arduino MKR WAN 1300 antes de descartar la tecnología por completo. Además, tratándose de una tecnología tan joven, no se pueden hacer afirmaciones seguras sobre lo que el futuro le depara al protocolo LoRaWAN.

Capítulo 8. INTEGRACIÓN CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE

En este capítulo final se va a explicar cómo este trabajo está alineado con los Objetivos de Desarrollo Sostenible (ODS). Para ello se va a introducir primero el ODS con el que más se alinea este proyecto.

8.1 CONTEXTUALIZACIÓN

Los Objetivos de Desarrollo Sostenible, también conocidos como Objetivos Mundiales, tienen su origen en el año 2015, cuando los Estados Miembros de la Organización de las Naciones Unidas adoptaron la "Agenda 2030 para el Desarrollo Sostenible" a fin de poner fin a la pobreza, proteger el planeta y asegurar la paz y la prosperidad para todas las personas. Con esta agenda, se incorporaron al Programa de las Naciones Unidas para el Desarrollo (PNUD) 17 objetivos que establecen que establecen que para conseguir un equilibrio de sostenibilidad medio ambiental, económico y social, es fundamental aceptar que las acciones en un territorio siempre van a tener consecuencias en otro, y viceversa. Son objetivos globales que buscan la colaboración de todas las personas para conseguir el ambicioso cambio que plantea el PNUD.

El objetivo con el que principalmente se alinea este trabajo es el número 7 "Garantizar el acceso a una energía asequible, y no contaminante". Este objetivo surge para combatir problemas como que 789 millones de personas no tuviesen acceso a electricidad en 2018 (Nations, 2021). Persigue estimular el consumo de energías renovables, que en 2017 representaba solo un 17 % del consumo total (Nations, 2021) y mejorar el acceso de energía asequible y fiable para instalaciones sanitarias ya que en 2018 uno de cada cuatro hospitales no contaba con este acceso (Nations, 2021). Para lograr este objetivo global, se han establecido 4 metas que se pretenden alcanzar en 2030 y que pueden servir como

indicadores del estado de este ODS. Las metas se pueden encontrar en la página de las Naciones Unidas y son los siguientes (Nations, 2021):

- 7.1 De aquí a 2030, garantizar el acceso universal a servicios energéticos asequibles, fiables y modernos (Nations, 2021)
- 7.2 De aquí a 2030, aumentar considerablemente la proporción de energía renovable en el conjunto de fuentes energéticas (Nations, 2021)
- 7.3 De aquí a 2030, duplicar la tasa mundial de mejora de la eficiencia energética (Nations, 2021)
- 7.a De aquí a 2030, aumentar la cooperación internacional para facilitar el acceso a la investigación y la tecnología relativas a la energía limpia, incluidas las fuentes renovables, la eficiencia energética y las tecnologías avanzadas y menos contaminantes de combustibles fósiles, y promover la inversión en infraestructura energética y tecnologías limpias (Nations, 2021)
- 7.b De aquí a 2030, ampliar la infraestructura y mejorar la tecnología para prestar servicios energéticos modernos y sostenibles para todos en los países en desarrollo, en particular los países menos adelantados, los pequeños Estados insulares en desarrollo y los países en desarrollo sin litoral, en consonancia con sus respectivos programas de apoyo (Nations, 2021)

8.2 ALINEACIÓN CON EL ODS Nº 7

Al tratarse de un trabajo de investigación y no de un proyecto que proponga una implantación real de algo, puede ser difícil entender cómo se alinea con el objetivo de "Garantizar el acceso a una energía asequible, y no contaminante". Sin embargo, este trabajo se encuentra alineado de manera directa con el ODS Nº 7 por dos motivos principales.

El primero, es por el uso de los TENGs. Los TENGs son fuentes de generación de energía eléctrica limpia. No generan contaminación directa en su uso y pueden ser usados para generar energía eléctrica que almacenar en baterías (Zi, y otros, 2016) o, como se ha

analizado en este trabajo, ser utilizados como sensores autoalimentados sustituyendo así otro tipo de sensores con consumo de energía y mayor contaminación. Otro motivo por el que los TENG están alineados con el ODS en cuestión, es que tienen un coste de fabricación muy bajo. Si se comparan con los PENG, la diferencia es notable (Vivekananthan, y otros, 2020). De esta manera los TENG se alinean de doble manera con el ODS, proporcionando energía limpia que además es accesible. Cuanto más se investigue acerca de estos nano generadores, más se acerca el día en el que puedan contribuir de manera global a las metas que se establecen desde el PNUD para 2030.

El segundo es la propia tecnología LoRaWAN se basa en dispositivos con muy poco consumo energético. Además, la transmisión de información se hace por radio frecuencia lo que reduce la contaminación que puede generar un cableado. Con sistemas de comunicación que no requieran de mucho consumo energético y eviten la producción de grandes baterías que resultan muy difíciles de reciclar consigue el progreso hacia un mundo conectado sin perjudicar el equilibrio de sostenibilidad medio ambiental, económico y social.

8.3 CUANTIFICACIÓN

Al tratarse de un proyecto de investigación sin una intención de implantación clara a día de hoy, lamentablemente no es posible realizar una cuantificación de cómo podría afectar el desarrollo y establecimiento de esta tecnología.

Capítulo 9. BIBLIOGRAFÍA

Allan, A. (2019). *A New LoRaWAN Distance Record*. Obtenido de Medium: https://aallan.medium.com/a-new-lorawan-distance-record-577c8bc11d7b (Fecha de último acceso: 09/07/2021)

Arduino. (2021). *ARDUINO STORE*. Obtenido de https://store.arduino.cc/arduino-mkr-wan-1300-lora-connectivity-1414 (Fecha de último acceso: 09/06/2021)

Augustin, A., Yi, J., Clausen, T., & Townsley, W. M. (2016). A Study of LoRa: Long Range & Low Power Networks for the Internet of Things. *Sensors*, 16(9), 1466. MDPI AG. Retrieved from http://dx.doi.org/10.3390/s16091466

Eric, B. (2018). *Lora-Lora Documentation*. Obtenido de https://www.google.com/search?q=lorawan+fair+use+policy&oq=lorawan+fair+use&aqs=chrome.1.69i57j0i22i30j69i60.5473j0j15&sourceid=chrome&ie=UTF-8 (Fecha de último acceso: 09/07/2021)

Cheng, T., Gao, Q., & Wang, Z. L. (2019). The Current Development and Future Outlook of Triboelectric Nanogenerators: A Survey of Literature. *Advanced Materials Technologies*, 4, 1800588. Retrieved from https://doi.org/10.1002/admt.201800588

Haxhibeqiri, J., Poorter, E. D., Moerman, I., & Hoebeke, J. (2018). A Survey of LoRaWAN for IoT: From Technology to Application. *Sensors*, 18, 3995. Retrieved from https://doi.org/10.3390/s18113995

Jie, Y., Jia, X., Zou, J., Chen, Y., Wang, N., Wang, Z. L., & Cao, X. (2018). Natural Leaf Made Triboelectric Nanogenerator for Harvesting Environmental Mechanical Energy. *Advanced Energy Materials*, 8, 1703133. Retrieved from https://doi.org/10.1002/aenm.201703133

Kim, D. W., Lee, J. H., Kim, J. K., & Jeong, U. (2020). Material aspects of triboelectric energy generation and sensors. *NPG Asia Materials*, 12, 6. Retrieved from https://doi.org/10.1038/s41427-019-0176-0

Liu, W., Wang, Z., Wang, G., Liu, G., Chen, J., Pu, X., . . . Wang, Z. L. (2019). Integrated charge excitation triboelectric nanogenerator. *Nat Commun* 10, 1426. Retrieved from https://doi.org/10.1038/s41467-019-09464-8.

LoRaWAN, P. o. (2017). *LoRaWAN*. Obtenido de https://lorawan.es/ (Fecha de último acceso: 25/05/2021)

Nations, U. (2021). *Department of Economic and Social Affairs*. Obtenido de https://sdgs.un.org/goals/goal7 (Fecha de último acceso: 11/05/2021)

RAKWireless. (2014). *RAKWireless Store*. Obtenido de https://store.rakwireless.com/products/rak831-gateway-module?variant=39942881083590 (Fecha de último acceso: 23/06/2021)

RaspberryPi. (2015). *RaspberryPi Products*. Obtenido de https://www.raspberrypi.org/products/raspberry-pi-2-model-b/ (Fecha de último acceso: 23/06/2021)

Sabas. (2017). *Medium.* Obtenido de Haciendo IoT con LoRa: https://medium.com/beelan/haciendo-iot-con-lora-capitulo-2-tipos-y-clases-de-nodos-3856aba0e5be (Fecha de último acceso: 23/06/2021)

Store, A. (2021). *Store Home-Arduino MKR WAN 1300 (LoRa connectivity)*. Obtenido de https://store.arduino.cc/arduino-mkr-wan-1300-lora-connectivity-1414 (Fecha de último acceso: 23/06/2021)

TheThingsNetwork. (2021). *The Things Network.* Obtenido de https://www.thethingsnetwork.org/ (Fecha de último acceso: 06/06/2021)

TheThingsNetworkNetworkArchitecture. (2021). *The Things Network*. Obtenido de https://www.thethingsnetwork.org/docs/network/architecture/ (Fecha de último acceso: 09/06/2021)

TheThingsNetworkOverview. (2021). *The Things Network*. Obtenido de https://www.thethingsnetwork.org/docs/network/ (Fecha de último acceso: 09/06/2021)

Visser, H. (2021). *The Things Network*. Obtenido de https://www.thethingsnetwork.org/article/setting-up-a-private-handler-connected-to-the-public-community-network (Fecha de último acceso: 06/06/2021)

Vivekananthan, V., Chandrasekha, A., N. R., Purusothaman, Y., Khandelwal, G., & Kim, S.-J. (2020). Triboelectric Nanogenerators: Design, Fabrication, Energy Harvesting, and Portable-Wearable Applications. *IntechOpen* 10.5772/intechopen.90951

Wu, C., Wang, A. C., Ding, W., Guo, H., & Wang, Z. L. (2019). Triboelectric Nanogenerator: A Foundation of the Energy for the New Era. *Advanced Energy Materials*, 9, 1802906. Retrieved from https://doi.org/10.1002/aenm.201802906

Zi, Y., Wang, J., Wang, S., Li, S., Wen, Z., Guo, H., & Wang, Z. L. (2016). Effective energy storage from a triboelectric nanogenerator. *Nat Commun*, 7, 10987. Retrieved from https://doi.org/10.1038/ncomms10987

ANEXO I. CÓDIGO FUENTE

NODOS FINALES

REGISTRO EN THE THINGS NETWORK

Registrar dispositivo

```
#include <MKRWAN.h>

// Select your region (AS923, AU915, EU868, KR920, IN865, US915, US915_HYBRID)
_lora_band region = EU868;

LoRaModem modem(Serial1);

void setup() {
  Serial.begin(115200);
  while (!Serial);
  if (!modem.begin(region)) {
    Serial.println("Failed to start module");
    while (1) {}
  };
  Serial.print("Your device EUI is: ");
  Serial.println(modem.deviceEUI());
}

void loop() {
}
```

Decodificadores

TEXTO

```
function Decoder(bytes, port) {
  // Decode plain text; for testing only
  return {
      ALARMSTATUS: String.fromCharCode.apply(null, bytes)
  };
}
```

PULSO

```
function Decoder(bytes, port) {
var decoded = {};
decoded.value10 = bytes[0];

decoded.value11 = bytes[1];

decoded.value12 = bytes[2];

decoded.value13 = bytes[3];

decoded.value14 = bytes[4];

decoded.value15 = bytes[5];

decoded.value16 = bytes[6];

decoded.value17 = bytes[7];

decoded.value18 = bytes[8];

decoded.value19 = bytes[9];

decoded.value20 = bytes[10];

decoded.value21 = bytes[11];

decoded.value22 = bytes[12];

decoded.value23 = bytes[13];

decoded.value24 = bytes[14];

decoded.value25 = bytes[15];

decoded.value26 = bytes[16];

decoded.value27 = bytes[17];

decoded.value28 = bytes[18];

decoded.value29 = bytes[19];

decoded.value30 = bytes[20];

decoded.value31 = bytes[21];

decoded.value32 = bytes[22];

decoded.value33 = bytes[23];

decoded.value34 = bytes[24];
```

```
decoded.value35 = bytes[25];

decoded.value36 = bytes[26];

decoded.value37 = bytes[27];

decoded.value38 = bytes[28];

decoded.value39 = bytes[29];

decoded.value40 = bytes[30];

decoded.value41 = bytes[31];

decoded.value42 = bytes[32];

decoded.value43 = bytes[33];

decoded.value44 = bytes[34];

decoded.value45 = bytes[35];

decoded.value46 = bytes[36];

decoded.value47 = bytes[37];

decoded.value48 = bytes[38];

decoded.value49 = bytes[39];

decoded.value50 = bytes[40];

decoded.value51 = bytes[41];

decoded.value52 = bytes[42];

decoded.value53 = bytes[43];

decoded.value54 = bytes[44];

decoded.value55 = bytes[45];

decoded.value56 = bytes[46];

decoded.value57 = bytes[47];

decoded.value58 = bytes[48];

decoded.value59 = bytes[49];

decoded.value60 = bytes[50];

decoded.value61 = bytes[51];
```

```
decoded.value62 = bytes[52];

decoded.value63 = bytes[53];

decoded.value64 = bytes[54];

decoded.value65 = bytes[55];

decoded.value66 = bytes[56];

decoded.value67 = bytes[57];

decoded.value68 = bytes[58];

decoded.value69 = bytes[59];

decoded.value70 = bytes[60];

decoded.value71 = bytes[61];

decoded.value72 = bytes[62];

decoded.periodo = bytes[63];

return decoded;

}
```

FUNCIONALIDADES

TRANSMISIÓN DE UN MENSAJE DE ALARMA POR ACTIVACIÓN DE TENG

Programa

```
#include <MKRWAN.h>
#include "arduino_secrets.h"

int sensorPin1 = A1;
int sensorValue = 0;


// Select your region (AS923, AU915, EU868, KR920, IN865, US915, US915_HYBRID)
_lora_band region = EU868;

LoRaModem modem(Serial1);

void setup() {
  Serial.begin(115200);
  while (!Serial);
  if (!modem.begin(region)) {
```

```
    Serial.println("Failed to start module");
    while (1) {}
  };
  Serial.print("Your device EUI is: ");
  Serial.println(modem.deviceEUI());

  int connected = modem.joinOTAA(appEui, appKey);
  if (!connected) {
    Serial.println("Something went wrong; are you indoor? Move near a window and
retry");
    while (1) {}
  }
  Serial.println("Successfully joined the network!");

  Serial.println("Enabling ADR and setting low spreading factor");
  modem.setADR(true);
  modem.dataRate(5);
}

void loop() {

    sensorValue = analogRead(sensorPin1);
    Serial.println(sensorValue);

    if (sensorValue>=800) {
      modem.beginPacket();
      modem.print("DANGER");



      int err = modem.endPacket(false);

      if (err > 0) {
        Serial.println("Big success!");
        Serial.println(sensorValue);  //Muestra el resultado en el monitor serial
      } else {



        Serial.println("Error");
      }
      delay(1000 * 2);
   }

}
```

TRANSMISIÓN DE VARIOS PULSOS DIGITALIZADOS

Programa

```
#include <MKRWAN.h>
#include "arduino_secrets.h"

int sensorPin1 = A1;
byte payload[64];
int sensorValue[63];
int scaledValue[63];

// Select your region (AS923, AU915, EU868, KR920, IN865, US915, US915_HYBRID)
_lora_band region = EU868;

LoRaModem modem(Serial1);

void setup() {
  Serial.begin(115200);
  while (!Serial);
  if (!modem.begin(region)) {
    Serial.println("Failed to start module");
    while (1) {}
  };
  Serial.print("Your device EUI is: ");
  Serial.println(modem.deviceEUI());

  int connected = modem.joinOTAA(appEui, appKey);
  if (!connected) {
    Serial.println("Something went wrong; are you indoor? Move near a window and
retry");
    while (1) {}
  }
  Serial.println("Successfully joined the network!");

  Serial.println("Enabling ADR and setting low spreading factor");
  modem.setADR(true);
  modem.dataRate(5);
}

void loop() {

unsigned long inicio, fin, transcurrido;
  int i;

    inicio=millis();  //Consultar ms fin del sketch: cada 25 ms. 30 datos

    for(i=0;i<63;i++){

      sensorValue[i] = analogRead(sensorPin1);
      scaledValue[i] = sensorValue[i]/4;
      payload[i] = scaledValue[i];
```

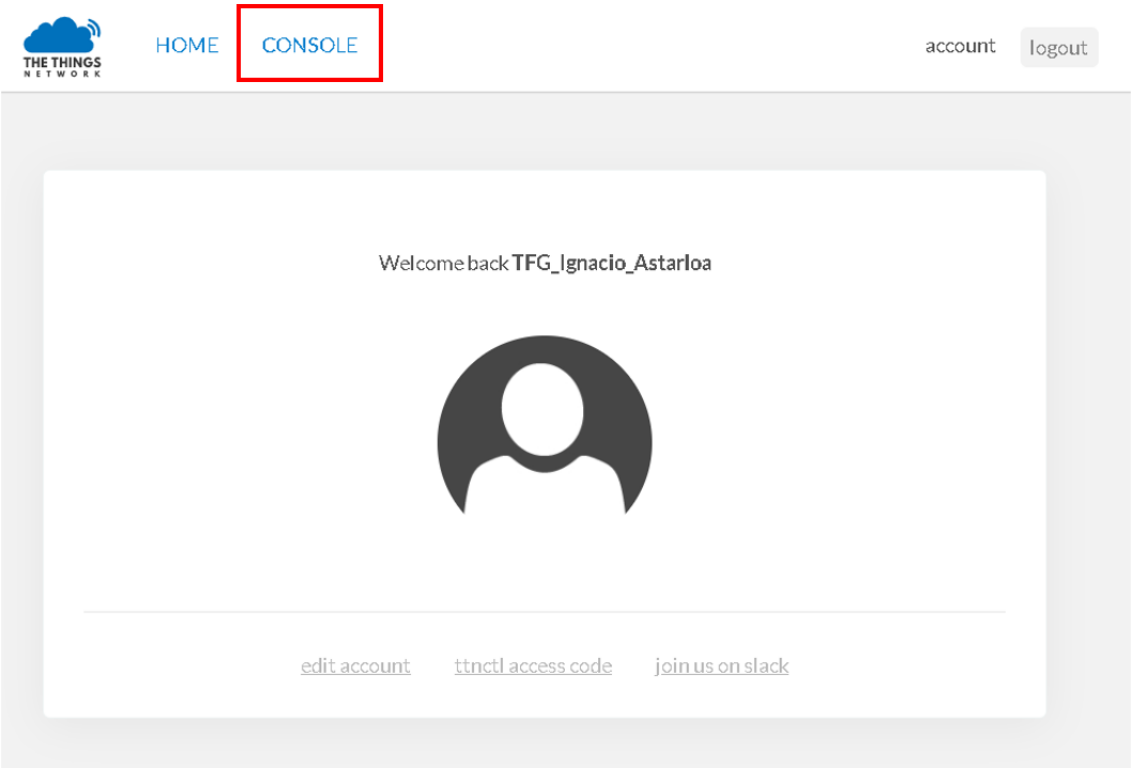
```
      delay(2);
    }

    fin=millis();  //Consultar ms fin del sketch
    transcurrido=fin-inicio;  //Calcula el tiempo desde la última lectura

    payload[63]=transcurrido;


    modem.beginPacket();     //LINEAS ORIGINALES
    modem.write(payload, sizeof(payload));     //LINEAS ORIGINALES



    int err = modem.endPacket(false);

    if (err > 0) {
      Serial.println("Big success!");
      Serial.println(transcurrido);  //Muestra el resultado en el monitor serial
    } else {
      Serial.println("Error");
    }
    delay(1000 * 6);
  //}
}
```

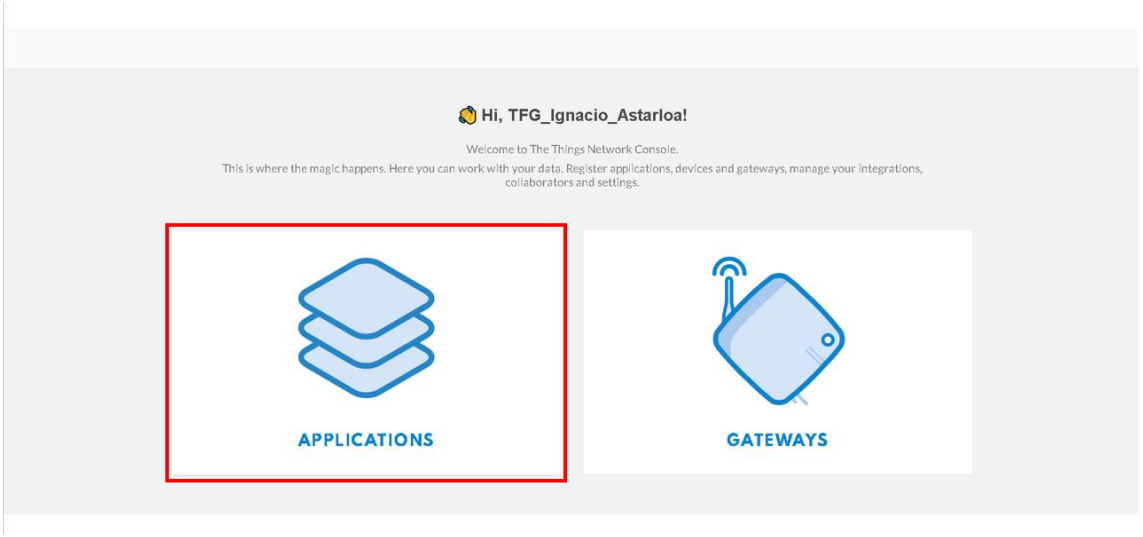
ANEXO II. PANTALLAZOS ILUSTRATIVOS

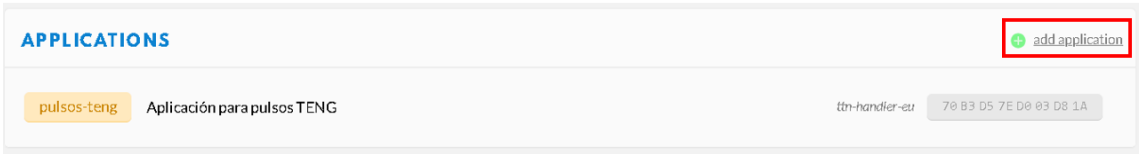
NODOS FINALES

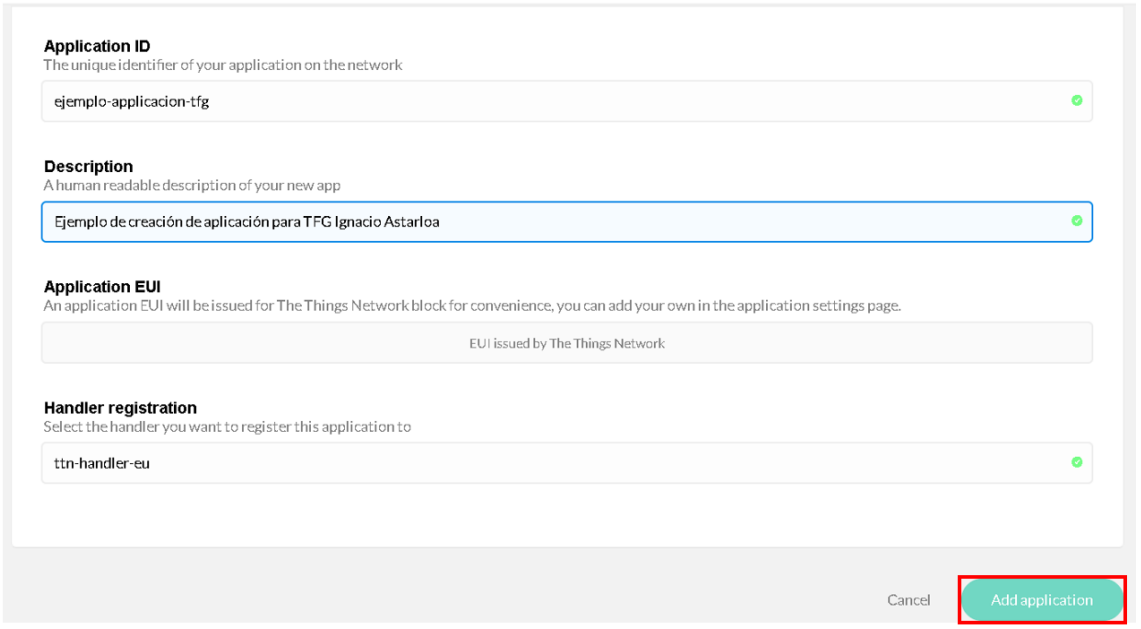
REGISTRO EN THE THINGS NETWORK

Creación de aplicación

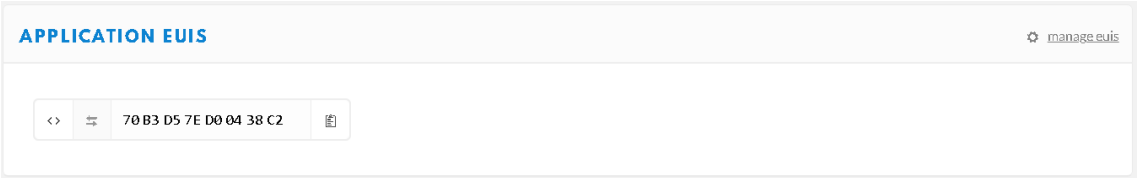



Hi, TFG_Ignacio_Astarloa!
Welcome to The Things Network Console.
This is where the magic happens. Here you can work with your data. Register applications, devices and gateways, manage your integrations, collaborators and settings.
APPLICATIONS
GATEWAYS

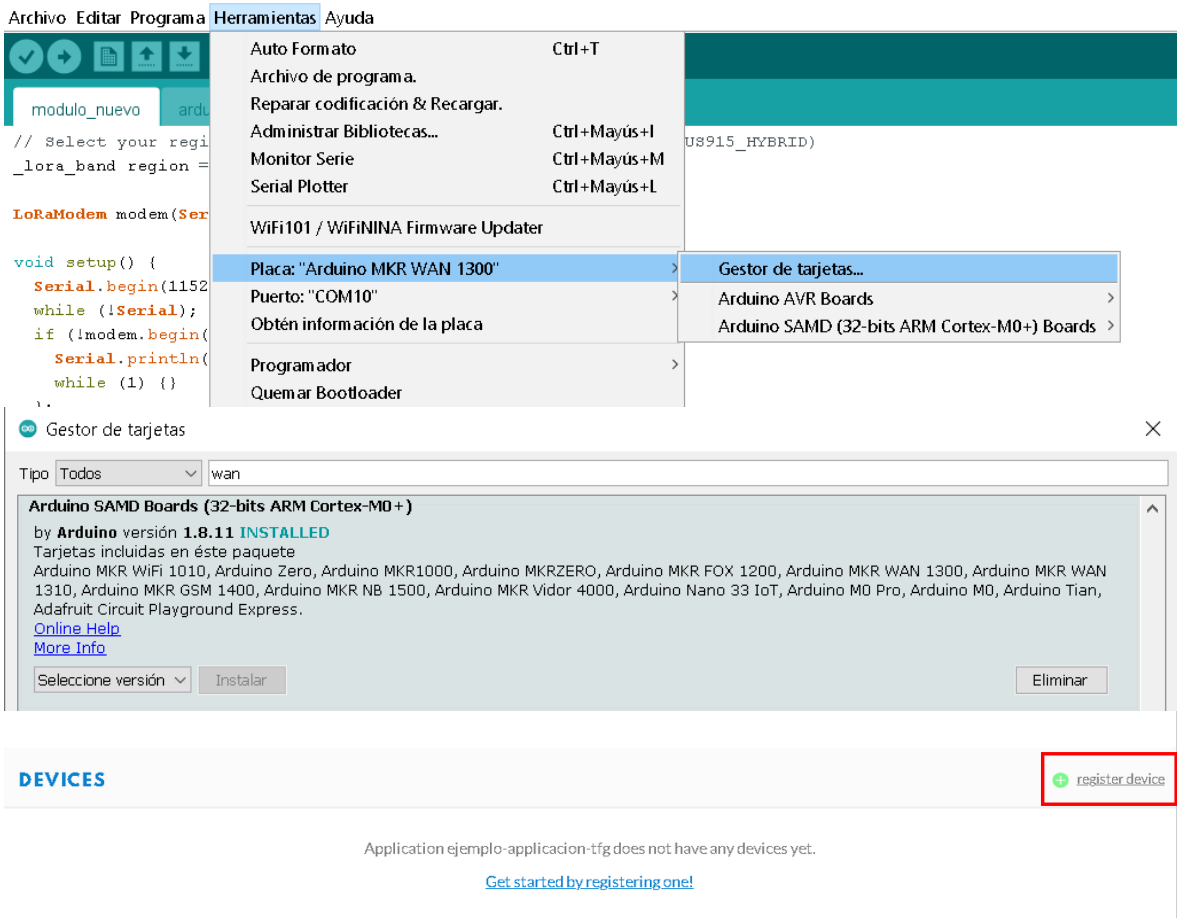

APPLICATIONS
add application
pulsos-teng
Aplicación para pulsos TENG
ttn-handler-eu
70 B3 D5 7E D0 03 D8 1A

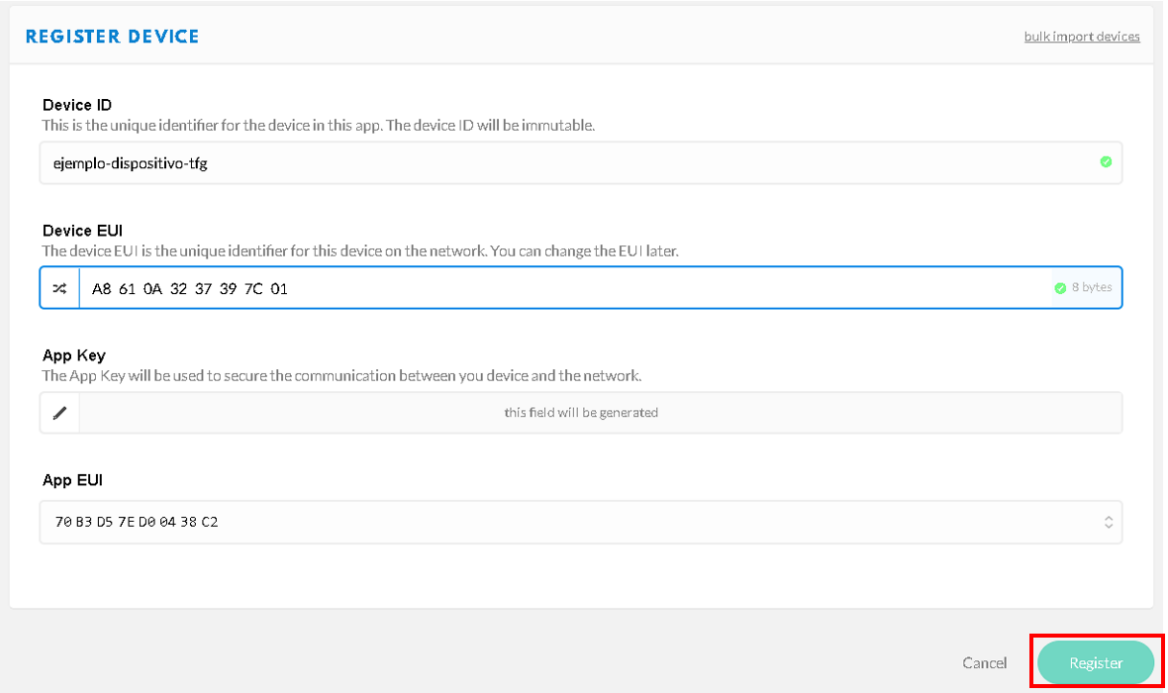

Application ID
The unique identifier of your application on the network
ejemplo-applicacion-tfg
Description
A human readable description of your new app
Ejemplo de creación de aplicación para TFG Ignacio Astarloa
Application EUI
An application EUI will be issued for The Things Network block for convenience, you can add your own in the application settings page.
EUI issued by The Things Network
Handler registration
Select the handler you want to register this application to
ttn-handler-eu
Cancel
Add application


ACCESS KEYS
manage keys
default key
devices
messages
base64

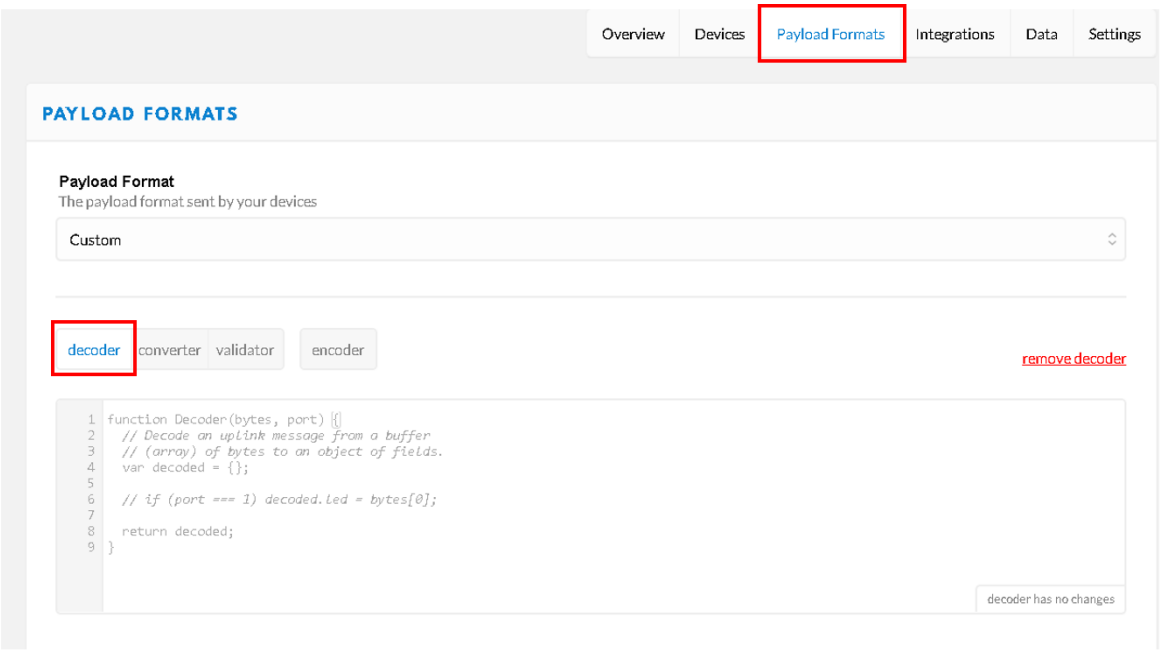


Registrar dispositivo

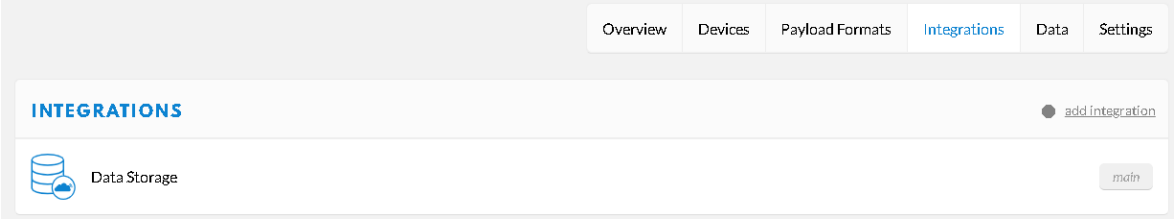




Decodificadores



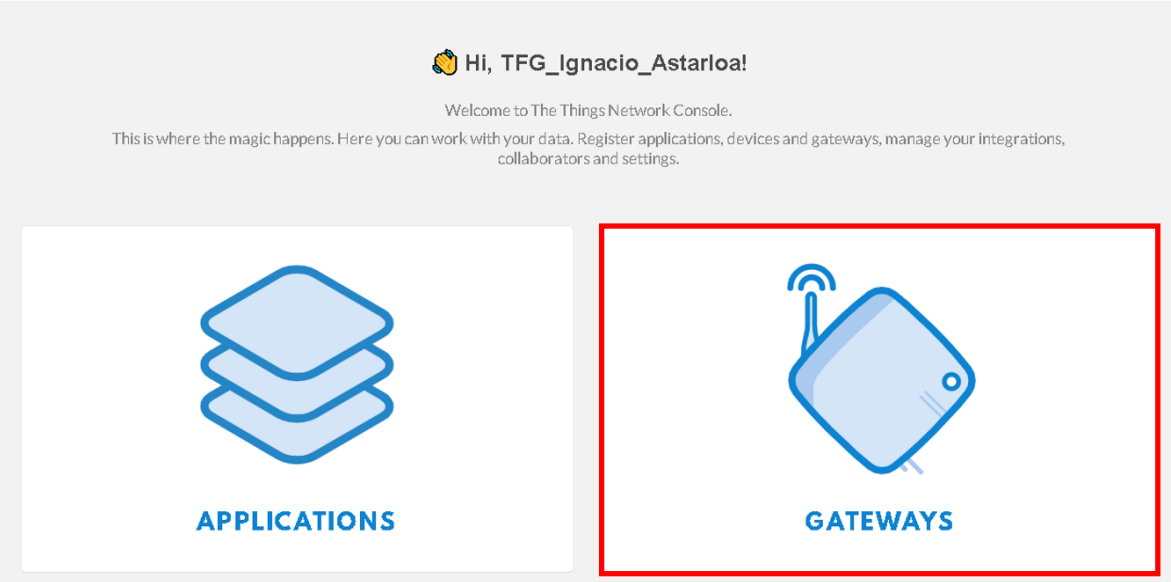
Almacenamiento de datos

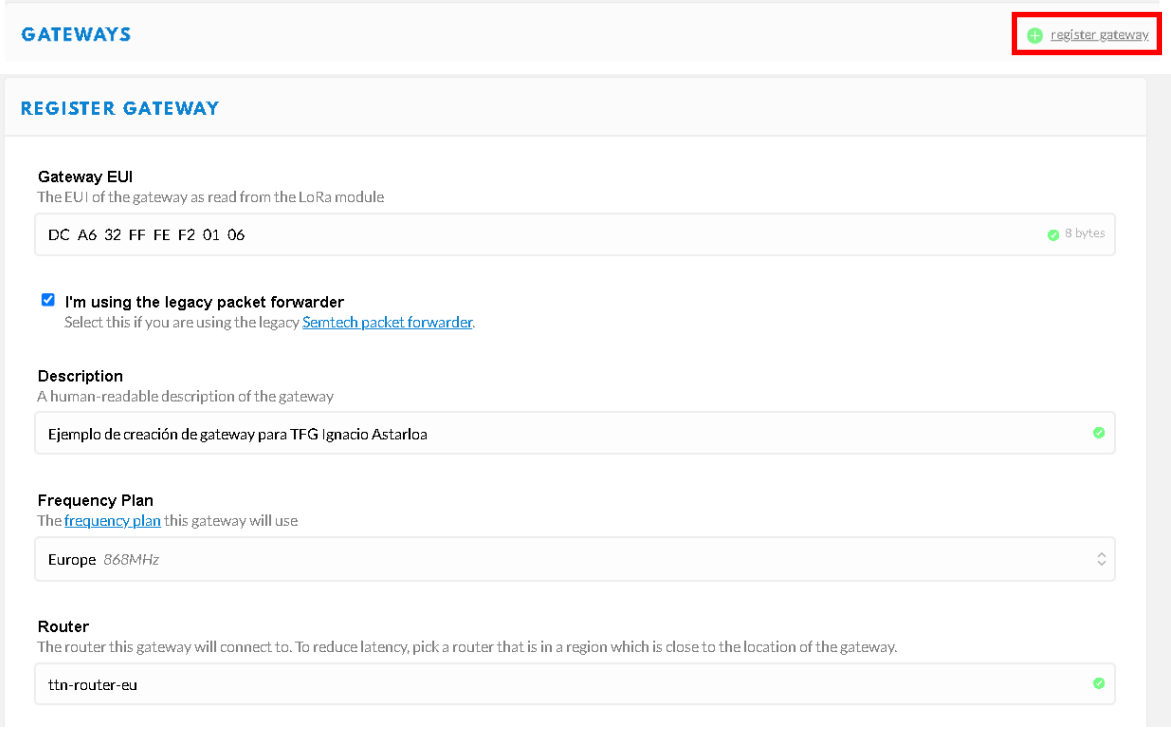


GATEWAYS

GATEWAY CON MÓDULO RAK 831 Y CONEXIÓN ETHERNET

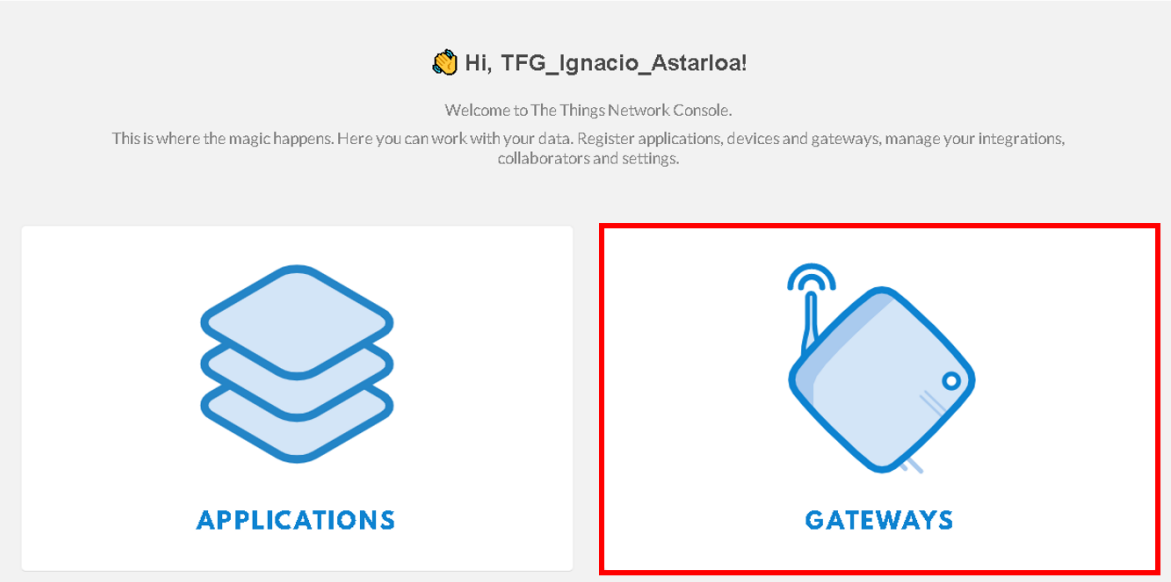
Registro en The Things Network

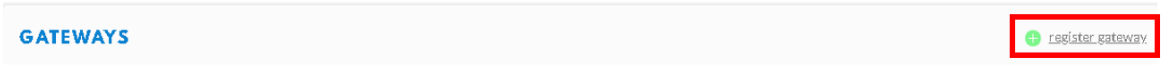


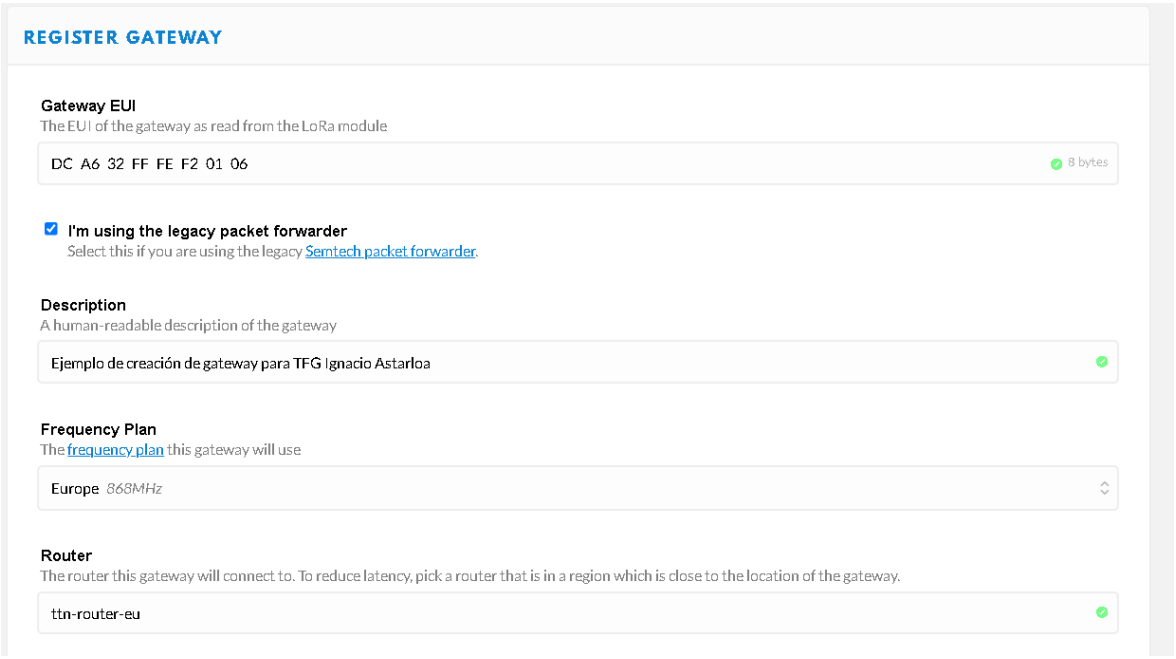


GATEWAY CON MÓDULO RAK 2245 Y CONEXIÓN WIFI

Registro en The Things Network







ANEXO III. ESPECIFICACIONES TÉCNICAS

MÁQUINAS USADAS PARA LA GENERACIÓN DE PULSOS

Q800 DMA – TA Instruments

Maximum Force	18 N
Minimum Force	0.0001 N
Force Resolution	0.00001 N
Strain Resolution	1 nanometer
Modulus Range	10^3 to 3×10^{12} PA
Modulus Precision	±1%
TanA Sensitivity	0.0001
TanA Resolution	0.00001
Frequency Range	0.01 to 200 Hz
Dynamic Sample Deformation Range	±0.5 to 10,000 pm
Temperature Range	-150 to 600 °C
Heating Rate	0.1 to 20 °C/min
Cooling Rate	0.1 to 10 °C/min
Isothermal Stability	±0.1 °C
Time/Temperature Superposition	Yes

E3000 Linear-Torsion All-Electric Dynamic Test Instrument - INSTRON

		34TM-5
Capacidad de fuerza	kN	5
	lbf	1125
Recorrido de la cruceta	mm	1172 (E1), 1651 (E2)
	in	46,1 (E1), 65,0 (E2)
Espacio de ensayo vertical (A)	mm	1242 (E1), 1744 (E2)
	in	48,9 (E1), 68,7 (E2)
Espacio de ensayo horizontal (B)	mm	420
	in	16,5
Velocidad máxima	mm/min	1016
	in/min	40
Velocidad mínima	mm/min	0,05
	in/min	0,002
Velocidad de retorno máxima	mm/min	1500
	in/min	59
Dimensiones (alt.× anch. × prof.)*	mm	1610 × 760 × 710
	in	63 × 30 × 28
Resolución de control de posición	nm	19,7
	µin	0,78
Rigidez axial del bastidor	kN/mm	38
	lb/in	217.000
Fuerza máxima a toda velocidad	kN	5
	lbf	1125
Velocidad máxima con toda la fuerza	mm/min	1016
	in/min	40
Peso	kg	122 (E1), 136 (E2)
	lb	268 (E1), 299 (E2)
Requisitos de potencia máxima	VA	730