

# Aplicaciones de LoRa y LoRaWAN con STM32WL55 Nucleo-64

Antonio Delgado Bejarano – adelgad01@us.es

**Resumen** – LoRa/LoRaWAN [1] es un estándar de comunicación que tiene una aplicación muy interesante en el campo de la salud conectada/IoMT. Sus principales características (largo alcance y consumo muy bajo de potencia) convierten a este estándar de comunicación en un buen candidato para aplicaciones como redes de sensores o monitorización, como se ha comprobado al consultar la literatura actual en cuanto al uso de LoRa/LoRaWAN en e-health. También se han revisado diferentes aplicaciones de LoRa/LoRaWAN con dos dispositivos STM32WL55 Nucleo-64. Se han estudiado las capacidades del dispositivo, así como el software asociado y las distintas aplicaciones, demos y ejemplos disponibles, llevando a la práctica dos de ellos.

**Índice de Términos** – IoMT, LoRa, LoRaWAN, Salud conectada, STM32WL55, STM Cube Programmer, STM Cube Monitor, The Things Networks

## I. INTRODUCCION

**E**n este documento presentaremos el Trabajo de Curso de la asignatura Tecnologías de Comunicación en Biomedicina de 2º curso del Máster en Ingeniería de Telecomunicación.

El trabajo tiene como motivación previa la realización para la asignatura de una actividad sobre estándares de comunicación para salud conectada / IoMT (Internet of Medical Things), en la que pude conocer y familiarizarme con el estándar LoRa/LoRaWAN. Partiendo de este punto se han realizado dos tareas:

En primer lugar, he consultado algunos artículos de investigación actuales para poder conocer distintas propuestas de uso de LoRa/LoRaWAN en salud conectada. Esto sirvió para poder determinar en qué situaciones o entornos es especialmente recomendable este estándar frente a otros también muy utilizados en el ámbito de la salud conectada, como Sigfox, Zigbee o las comunicaciones celulares.

En segundo lugar, se ha trabajado con dos dispositivos STM32WL55 Nucleo-64 para estudiar sus características y sus posibles aplicaciones. Este dispositivo permite trabajar con diferentes modulaciones radio en la banda Sub-GHz, entre ellas LoRa. Además, cuenta con un amplio abanico de entornos software que se pueden descargar para trabajar con él, así como con varios archivos pertenecientes a ejemplos, aplicaciones o demostraciones que pueden permitir poner en marcha una aplicación real.

Con todas estas herramientas (el dispositivo, el software asociado y los ejemplos de aplicación) se ha conseguido poner en marcha una demostración de red LPWAN (Low Power Wide Area Network) y se ha realizado una caracterización experimental de la comunicación con las herramientas disponibles. Para la generación de gráficos que permitan una mejor comprensión de los datos de la comunicación se ha generado también un script en MATLAB. Se describirá la demostración de red LPWAN probada y se estudiarán los resultados obtenidos.

Por último, dejaremos constancia de propuestas de futuras líneas de trabajo con dispositivos como los utilizados en el ámbito de las comunicaciones en IoMT, las cuales no han podido ser abordadas por limitaciones hardware pero son realmente interesantes.

## II. REPASO DE LORA / LORAWAN

Comenzaremos repasando las características del estándar LoRa/LoRaWAN que ya presentamos en clase en una de las actividades de la asignatura.

LoRa/LoRaWAN es un estándar para redes LPWAN [1]. Tiene varios competidores como Sigfox, Zigbee o NB-IoT. LoRa/LoRaWAN tiene dos características fundamentales: largo alcance y bajo consumo, las cuales lo convierten en una tecnología ideal para redes de dispositivos IoT en distintos contextos como industria, Smart Cities, logística, etc. En el contexto de la asignatura (ligado a la salud conectada) este tipo de estándares son muy útiles para monitorización y redes de sensores (de glucosa, por ejemplo), donde la eficiencia es fundamental, ya que habitualmente no cuentan con conexión a la corriente.

LoRa fue desarrollado por Semtech en 2009 y en 2015 se fundó la LoRa Alliance (donde intervienen empresas como Orange, Cisco o IBM) [2]. La LoRa Alliance se encarga de desarrollar las versiones del estándar y de certificar los dispositivos que emplean esta tecnología.

#### A. LoRa

LoRa hace referencia a la capa física sobre la que puede funcionar LoRaWAN. En una equivalencia con modelo OSI, LoRa sería equivalente a la primera capa y LoRaWAN sería equivalente a las capas 2 y 3, ya que funciona a nivel de enlace y red coordinando los parámetros de la conexión (ancho de banda, cifrado, canal, seguridad...) de la capa física (LoRa) de los dispositivos:

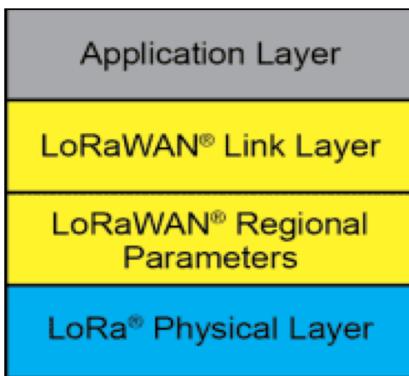


Fig. 1. Equivalencia entre LoRa/LoRaWAN y las capas del modelo OSI.

No es un requisito que LoRaWAN funcione sobre LoRa, ya que se pueden emplear otras capas físicas como FSK. Sin embargo, como podemos imaginar, lo habitual es que funcione sobre LoRa. Las dos principales características de LoRa son las siguientes:

LoRa permite comunicaciones punto a punto a gran distancia (más de 20 km en algunos casos), de forma que con una única puerta de enlace podríamos dar soporte a una red LoRaWAN en una ciudad entera. La red Proximus de Bélgica es un ejemplo de esto, ya que se ha podido cubrir gran parte del país con poca inversión en infraestructura gracias a las bondades de la tecnología LoRa/LoRaWAN. Sí que hay que garantizar que exista línea de visión directa entre puntos centrales de la red, o al menos que no existan obstáculos de gran consideración como colinas, ya que por la naturaleza de LoRa la potencia de transmisión es muy limitada y no permitirá salvar grandes obstáculos para comunicaciones a grandes distancias.

El otro pilar en el que se sostiene LoRa es en la eficiencia, permitiendo el uso masivo de dispositivos muy sencillos que consuman muy poca energía. Se pretende que las baterías alcancen una vida útil de más de diez años utilizando LoRa como capa física [3].

Para conseguir este desempeño y permitir comunicaciones de largo alcance con un mínimo consumo de energía LoRa utiliza una modulación de espectro ensanchado por chirp, CSS, que permite modificar de manera lineal y continua la frecuencia de la portadora (chirp).

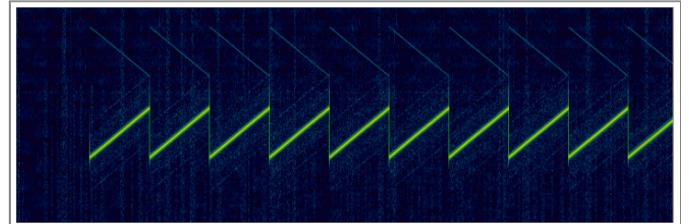


Fig. 2. Modulación CSS.

No entraremos en demasiado detalle sobre esto porque ya se describió en la tarea realizada sobre LoRa/LoRaWAN, aunque sí que mencionaremos el factor de ensanchamiento (SF), que define el número de bits usados para codificar un símbolo.

Una poderosa característica de LoRa es que valores diferentes de SF generan señales pseudo-ortogonales entre ellas, aunque tengan la misma frecuencia central y ancho de banda. Sería análogo a CDMA, ya que se transmiten al mismo tiempo señales en la misma frecuencia y con el mismo ancho de banda, pero con "códigos" diferentes. El SF está relacionado con la tasa de transferencia y con la inmunidad que alcanza la transmisión. Si SF crece aumenta la inmunidad a interferencias, pero a costa de disminuir la tasa de transferencia. Un valor alto de SF (10-12) permite que potencialmente lleguen más mensajes al destino a costa de reducir bastante la velocidad de transmisión.

CSS dota a LoRa de una gran inmunidad frente a interferencias obstáculos en interiores y efecto Doppler, permitiendo así las comunicaciones a grandes distancias. Otros datos de interés de los dispositivos LoRa son su baja sensibilidad (del orden de -140 dBm), una potencia de emisión muy pequeña para aumentar la eficiencia (de 14 a 20 dBm), un ancho de banda de entre 125 y 500 kHz y una tasa de transferencia también muy pequeña (de 0,3 a 50 kbps), motivos por los cuales consigue un consumo de energía muy comedido. El hecho de tener una tasa tan baja no es un problema para las aplicaciones masivas de IoMT donde los dispositivos transmiten muy pocos datos y además de manera infrecuente.

Por último, comentar que LoRa trabaja en la banda sub-GHz ISM (Industrial, Scientific & Medical). Más concretamente, las bandas de frecuencias de trabajo son 868 MHz en Europa, 915 MHz en América, y 433 MHz en Asia. Aunque son bandas no licenciadas sí que están reguladas por la ETSI y hay limitaciones en cuanto a, por ejemplo, la potencia de transmisión [2][3].

## B. LoRaWAN

Como hemos comentado, LoRaWAN incluye las capas de enlace y red, definiendo protocolos de comunicación y la arquitectura de la red [4]. Una red LoRaWAN consta de:

- **Dispositivos de nodo:** por lo general, estos dispositivos son sensores inalámbricos pequeños, a menudo alimentados por batería, que envían pequeñas cantidades de datos de vez en cuando. Los nodos funcionan con el método Aloha, es decir, se comunican por eventos cuando tienen datos listos para enviar, lo que permite ahorrar energía. Los dispositivos STM32WL55 que hemos usado son dispositivos de nodo.
- **Puertas de enlace/concentradores:** se pueden considerar como puntos de acceso. Contienen un módulo de radio LoRa y algún tipo de conexión IP. La conexión IP puede ser un cable Ethernet, conexión WiFi, conexión de datos 3G/4G o similar. Reciben la comunicación LoRa de los dispositivos del nodo y reenvían el mensaje a un servidor de red a través de su conexión IP. La idea es que una sola puerta de enlace pueda manejar miles de nodos y proporcionar cobertura de LoRaWAN en un área grande (normalmente áreas de centenares kilómetros).
- **Servidor de red:** Encargados principalmente del enrutamiento. Las puertas de enlace envían los mensajes que reciben al servidor de red que reenvía los mensajes al servidor de aplicaciones correcto. En el servidor de red se filtran paquetes, se eliminan paquetes redundantes, se lleva a cabo una adaptación de velocidad y todos los aspectos de verificación de seguridad. Es extremadamente importante que cualquier LPWAN incorpore seguridad (y más en caso de IoMT).
- **Servidor de aplicaciones:** Los mensajes de LoRaWAN terminan aquí, donde se almacenan o procesan. Corresponden a las aplicaciones IoT que usan los datos proporcionados por los nodos.

En cuanto a la arquitectura, LoRaWAN define una red de redes en estrella. Las redes en estrella permiten ahorrar batería en los nodos (dispositivos) porque se evita que un nodo tenga que reenviar información de otro nodo individual (como sí ocurre en las redes en malla). Por otra parte, cada puerta de enlace establece una pasarela entre el nodo individual y el servidor de red. Para conseguir largo alcance con una red en estrella las puertas de enlace deben poder manejar mucho volumen de mensajes de los nodos, como ya se ha comentado.

La puerta de enlace adapta su velocidad basándose en las ventajas de la modulación CSS de LoRa donde cambios en el factor de ensanchamiento provocan cambios en la velocidad de transmisión de los datos.

Aunque hemos comentado que en aplicaciones IoMT no es necesaria una gran tasa, si un nodo tiene un buen enlace y está cerca de una puerta de enlace, no hay razón para que siempre use la velocidad de datos más baja y llene el espectro disponible más tiempo del necesario. Al aumentar la velocidad de datos, el tiempo en el aire se acorta, lo que abre más espacio potencial para que otros nodos transmitan. La velocidad de datos adaptable también optimiza la vida útil de la batería de un nodo.

En el contexto de IoMT, LoRaWAN ofrece conexiones bidireccionales seguras, bajo consumo de energía (las capas de enlace y red son las que más consumen), largo alcance, soporte para movilidad y servicios de localización y bajo coste en equipamiento y al no usar bandas licenciadas.

Algunas soluciones como The Things Network proporcionan un ecosistema para crear nuevas redes de dispositivos IoT a nivel global mediante LoRaWAN. Uno de los ejemplos demostrativos que podemos encontrar con los dispositivos STM32WL55 es precisamente crear una red LoRaWAN usando The Things Networks. Lo comentaremos en secciones posteriores.

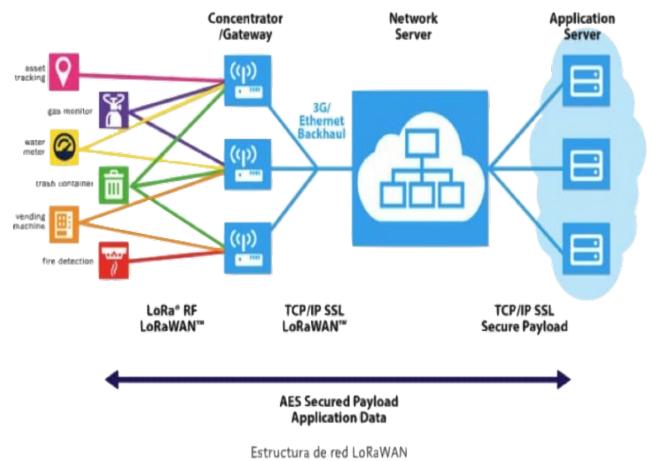


Fig. 3. Arquitectura de una red LoRaWAN [5].

## C. Clases de dispositivos.

Los nodos de LoRaWAN pueden trabajar de diferente forma, dando lugar a tres clases diferentes según el modo de funcionamiento [5]. Cada modo prioriza o bien la eficiencia en el consumo energético o bien la latencia de la comunicación:

- **Clase A:** Los dispositivos de clase A suelen ser sensores alimentados por batería, como un dispositivo de monitoreo de temperatura que envía la temperatura cada 15 minutos. En estos dispositivos la transmisión de enlace ascendente de cada dispositivo final es seguida por dos ventanas de recepción de enlace descendente cortas. Esta operación de Clase A es el sistema de dispositivo final de menor potencia

para aplicaciones que solo requieren comunicación de enlace descendente desde el servidor poco después de que el dispositivo final haya enviado una transmisión de enlace ascendente. El consumo de batería de estos dispositivos es muy bajo, ya que la interfaz radio está apagada la mayoría del tiempo. Sin embargo, la contrapartida es que la latencia es muy grande, ya que para recibir algún mensaje solo dispone de dos ventanas en los instantes posteriores a su transmisión.

- **Clase B:** Son dispositivos con ranuras de recepción programadas. Además de las ventanas de recepción aleatoria de Clase A, los dispositivos de Clase B abren ventanas de recepción adicionales en horarios programados establecidos por una baliza de programación enviada por las puertas de enlace. Esto es útil para dispositivos que necesitan poder enviar un mensaje de manera más confiable y frecuente. El ahorro de batería no es tan considerable, aunque reducimos considerablemente la latencia.
- **Clase C:** Estos dispositivos tienen su radio encendida todo el tiempo. Siempre están preparados para recibir información, excepto lógicamente cuando están transmitiendo un mensaje. Estos dispositivos suelen estar alimentados por la red eléctrica, ya que la interfaz radio está siempre encendida y el ahorro de energía es mínimo. Sin embargo, conseguimos una latencia tremadamente baja al poder transmitir y recibir información de manera instantánea cuando sea requerido.

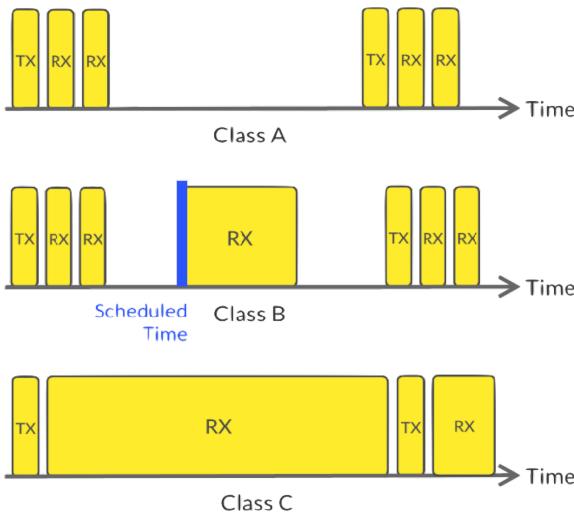


Fig. 4. Clases de dispositivos LoRa/LoRaWAN.

### III. APPLICACIONES CONCRETAS DE LORA/LORAWAN EN IOMT

En esta sección mencionaremos algunas aplicaciones de estos estándares de comunicación en aplicaciones de salud conectada. Para ello hemos buscado varios artículos de investigación actuales los cuales nos han permitido conocer el contexto de uso más favorable/recomendable para decantarnos por LoRa/LoRaWAN en el ámbito de la salud conectada.

En [6] se realiza una descripción del estándar similar a la que hemos hecho en la sección II y describen los resultados experimentales de un experimento realizado con el transceptor LoRa Hop-eRF RFM95/96 como nodo final. Como servidor de red utilizan el mencionado The Things Network y como Gateway (concentrador) usan una RaspberryPi3 con el transceptor LoRa HopeRF RFM95. El experimento realizado es sencillo: cuando el Gateway recibe un paquete LoRaWAN lo reenvía al servidor de red. El nodo final es de clase A y transmite tramas LoRa con una frecuencia y SF prefijados.

Con un SF de 4/5, un ancho de banda de 125 kHz y transmitiendo a 868.1 MHz en interiores miden el tiempo de vuelo de las tramas para diferentes longitudes de estas, obteniendo el siguiente resultado:

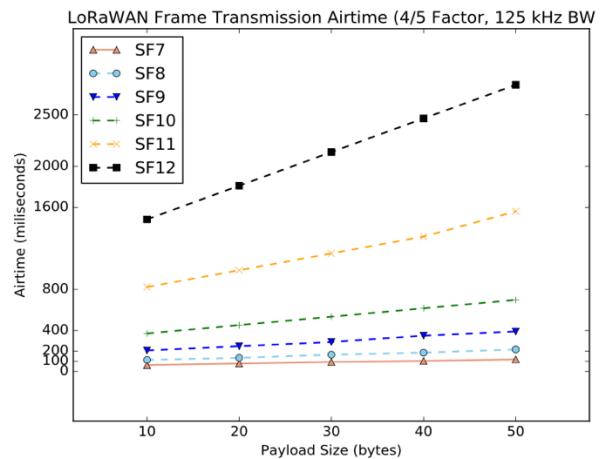


Fig. 5. Tiempo de vuelo de transmisión de tramas LoRaWAN [6]

En [6] detallan que, puesto que al aumentar el SF el tiempo de vuelo crece incluso para tramas pequeñas, LoRaWAN solo podría ser usada para servicios en tiempo real en casos donde el mensaje sea muy pequeño y el SF no exceda valores de 7 u 8. Con estas configuraciones sí que es posible usar LoRaWAN en escenarios de monitorización de temperatura o presión sanguínea, los cuales se pueden codificar en mensajes pequeños y pueden notificarse con una frecuencia de días u horas. También comparan el precio del equipamiento de LoRa con otras alternativas como BLE o Zigbee. Los dispositivos LoRa son más caros, aunque permiten una transmisión a mucha más distancia. Los nodos finales LoRa se pueden encontrar desde unos \$8, mientras que los concentradores pueden irse a \$200.

Otro artículo interesante es [7]. En dicho artículo proponen y caracterizan en cuanto a área cubierta y consumo de potencia un sistema de monitorización de presión sanguínea, glucosa y temperatura mediante sensores IoT y una red LoRaWAN. Los autores eligen LoRaWAN por su bajo consumo de batería en los nodos finales y la baja potencia de transmisión, además de la seguridad y eficiencia en la comunicación. La infraestructura LoRa cuenta con un Gateway y varios sensores que actúan como nodos transmisores, pero no mencionan qué servidor de red utilizan.

En primer lugar realizan una comparativa entre LoRaWAN y protocolos de comunicaciones celulares (sin incluir 3G/4G porque se escapan tanto en precio como en especificaciones de la aplicación que se está considerando):

	Average Range	Energy Consumption (mA)	Minimum Hardware Cost	Maximum Data Rate
<b>LoRa</b>	60 km	Idle mode: 2.8 mA	Transceiver: 10 \$	50 kbps (downlink)
		Continuous receive mode: 14.2 mA Send mode 38.9 mA	Gateway: 250 \$	50 kbps (uplink)
<b>GPRS</b>	60 km	Idle mode: 20 mA	Transceiver: 50 \$	85.6 kbps (downlink)
		Continuous receive mode: 130 mA Send mode 2000 mA	Gateway: 10000 \$	14 kbps (uplink)

Fig. 6. Comparativa de coste de equipamiento entre LoRaWAN y GPRS [7]

Como se aprecia, los dispositivos LoRa son bastante más baratos y mucho más eficientes en cuanto a consumo de energía. Además, a esto habría que sumar el coste de infraestructuras y bandas de frecuencia asociados a GPRS que no son considerados en LoRaWAN al transmitir en banda no licenciada. Por todo ello, en el artículo concluyen que LoRaWAN mejora las prestaciones de las comunicaciones celulares para redes de sensores como la que se está considerando. También presentan dos evaluaciones con las que pretenden determinar el área cubierta y el consumo de potencia.

Para determinar el área sitúan el concentrador a unos 12 metros de altura en una azotea y conducen en coche con el nodo final transmitiendo datos desde diferentes ubicaciones. Consiguieron unos 2 km de cobertura en zonas con muchos edificios y bloques de pisos y unos 3 km en zonas más despejadas.

Para cuantificar el consumo de potencia utilizaron un amperímetro que determinara el consumo de corriente al enviar datos y en reposo. El consumo en reposo es de 4.7 mA, muy por debajo de los 20 mA de una transmisión GSM en reposo. Al transmitir datos hay que considerar que la trama enviada es relativamente larga (93 bytes), ya que lleva información de presión sistólica, diastólica, pulso, glucosa en sangre y temperatura. El tiempo de envío es de 500 ms y el consumo de corriente al enviar un paquete LoRa es de 148 mA. Por tanto, el kit que utilizar consume unos 148 mA durante 500 ms y unos 4 mA el resto del tiempo que no está transmitiendo, por lo que con la batería de su dispositivo tienen para más de diez días de uso sin cargar el kit.

También se ha consultado el artículo [8] que propone utilizar LoRaWAN como método de comunicación inalámbrica entre hospitales. También utilizan un kit con un transmisor LoRa (en este caso a 915 MHz, ya que la aplicación se desarrolló en Chile) y un Gateway. Para caracterizar la cobertura realizan un estudio de la red LoRaWAN en Santiago de Chile, probando la conexión entre el hospital A y el hospital B (a 1 km de distancia) y entre el hospital A y el hospital C (a 8.5 km de distancia). Además, hay que tener en cuenta la peculiar topología de Chile, con numerosas zonas de colinas que dificultan la comunicación. También aportan información sobre la estructura de los paquetes intercambiados entre los hospitales y la aplicación final (basada en Android) para la consulta de los datos intercambiados.

Por último, mencionar brevemente el artículo [9], el cual aborda el problema de las personas mayores que viven en entornos rurales y que requieren de soluciones de poco coste y de fácil mantenimiento para su monitorización. LoRa encaja bastante bien en este propósito. Además, en entornos rurales hay ciertas peculiaridades como baja calidad de conexión a internet (las infraestructuras de 3G/4G pueden no cubrir determinadas áreas), casas muy espaciadas, posible falta de conexión a la corriente, etc. Por otra parte, las personas mayores que viven alejadas de hospitales no tienen la opción de cambiar o reparar el dispositivo que los monitoriza cada poco tiempo, por lo que aumentar la eficiencia y la vida útil del sensor es fundamental, lo cual casa perfectamente con los principios de LoRa. Utilizan The Things Networks para construir la red (como otros ejemplos presentados) y realizan una caracterización en cuanto a área y consumo similar a las ya vistas

#### IV. STM32WL55 NUCLEO 64

Una vez consultados artículos especializados y comprobado que LoRa/LoRaWAN es una alternativa muy útil en la mayoría de aplicaciones de salud conectada procederemos a estudiar el dispositivo facilitado para la realización de este trabajo, el STM32WL55.

##### A. Dispositivo

Se han empleado dos dispositivos STM32WL55 Nucleo 64 Board, los cuales permiten probar e implementar aplicaciones LoRa gracias al microcontrolador Wireless sub-GHz de ultra bajo consumo que incorpora [10]. Este microcontrolador es multiprotocolo e incorpora un transceptor de radiofrecuencia en el rango de 150MHz a 960MHz para modulaciones LoRa, FSK, GFSK, MSK, GMSK y BPSK. Cuenta también con una memoria flash de 256 Kbytes, 3 LEDs de usuario, 3 botones de usuario y uno de reset para interactuar con la placa y un depurador on-board embebido. Puede alimentarse conectando un USB a un ordenador o con un cargador de 5V. También cuenta con una serie de jumpers que hay que colocar en una

determinada configuración para cada aplicación concreta, por lo que los describiremos al describir la aplicación que hemos probado.

Cuenta también con la posibilidad de añadir extensiones con nuevas funcionalidades mediante Arduino Uno V3 y ST morpho headers. Este dispositivo se ofrece en dos variantes: la JC1 que cubre el rango de frecuencias de 865MHz a 928MHz y la JC2 que cubre el rango de 433MHz a 510MHz. En nuestro caso contamos con dos dispositivos de la variante JC1.

El layout de la placa tiene la siguiente descripción en cuanto a conectores y botones. Pueden apreciarse en la parte inferior los tres LEDs, los tres botones de usuario y el botón de reset, así como el microcontrolador, los conectores para Arduino y Morpho y el debugger:

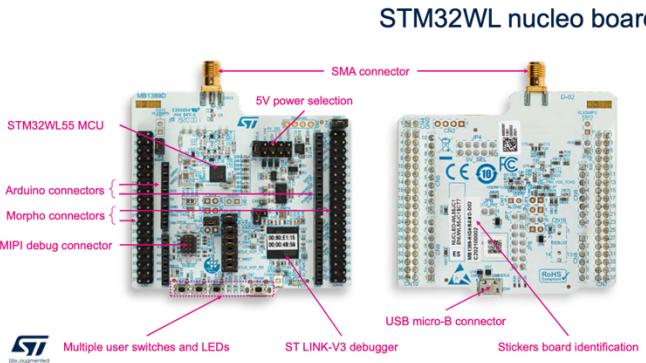


Fig. 7. Layout de la placa en vista superior [11]

Dentro de lo que sería una red LoRaWAN, el dispositivo probado corresponderá a un nodo final. STMicroelectronics también vende el Gateway para montar una aplicación LoRaWAN real. En cuanto al servidor de red podría usarse The Things Networks como hemos visto en otros muchos ejemplos de la literatura.

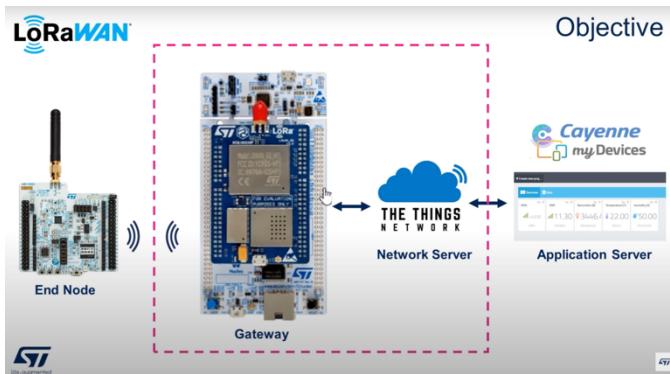


Fig. 8. Red LoRaWAN con dispositivos de STMicroelectronics y The Things Networks como servidor de red LoRaWAN [12]

El precio del nodo final es de unos 40€, mientras que el Gateway asciende hasta los 120€ [13].

### B. Software asociado

STMicroelectronics dispone de la iniciativa SMT32Cube que proporciona un conjunto de programas para simplificar la labor de los desarrolladores a la hora de desarrollar, probar y operar sus aplicaciones [14]. En concreto, se ha trabajado en mayor o menor medida con tres de los programas:

- SMT32CubeIDE, el cual proporciona un entorno de desarrollo basado en Eclipse para escribir código, compilarlo y depurarlo en la placa. La placa se programa en C y C#. No se ha utilizado finalmente porque la aplicación implementada no lo requería, pero sí se ha manejado para intentar probar otras aplicaciones que finalmente no hemos podido implementar por no disponer de un dispositivo Gateway. Esta disponible para Linux, MacOS y Windows en: <https://www.st.com/en/development-tools/stm32cubeide.html#get-software>
- STM32CubeProgrammer, que permite conectarse a la placa para cargar programas, consultar cada posición de la memoria, reiniciar el dispositivo a nivel de hardware y software y actualizar el firmware de este. Se ha utilizado en la aplicación probada. A priori funciona con los sistemas operativos presentados anteriormente, aunque en mi caso he tenido problemas de compatibilidad en MacOS, no así en Windows. Puede encontrarse en <https://www.google.com/search?client=safari&rls=en&q=stm+cube+programmer&ie=UTF-8&oe=UTF-8>

- STM32CubeMonitor, que permite monitorizar la aplicación mientras está ejecutándose. Funciona con flujos de trabajo y es bastante sencillo e intuitivo de utilizar. Igual que antes, solo he conseguido hacerlo funcionar en Windows. Disponible en: <https://www.google.com/search?client=safari&rls=en&q=stm+cube+monitor&ie=UTF-8&oe=UTF-8>

Mencionamos también los dos programas disponibles que no he necesitado o podido utilizar, caso del STM32CubeMonitorPower, que permite monitorizar el uso de energía del microcontrolador en tiempo real. Hubiera sido interesante utilizarlo, pero por motivos que desconozco no he conseguido hacerlo funcionar con la aplicación probada. El otro es el programa STM32CubeMX, un entorno gráfico para la generación automática de código C.

### C. Ejemplos y demostraciones

En la página del dispositivo [15] podemos encontrar un conjunto de aplicaciones, ejemplos y demostraciones para probar (dentro de la carpeta Projects):

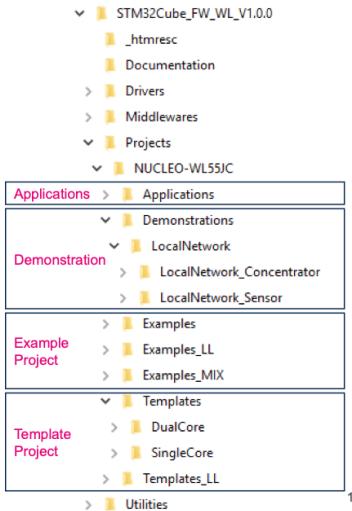


Fig. 9. Ejemplos y demostraciones para probar el STM32WL55 [15]

En algunos casos se proporciona el código para ser compilado y depurado en el IDE, mientras que en otros se proporcionan directamente los archivos binarios para cargarlos en la placa mediante el STM32CubeProgrammer. Hay ejemplos de múltiples modulaciones como LoRa o Sigfox. En el caso que nos ocupa, destacamos algunos de los ejemplos de LoRa que hay disponibles:

- LoRaWAN\_End\_Node:

Esta aplicación sirve para medir el nivel de batería y temperatura del microprocesador y enviarlos periódicamente a una red LoRaWAN utilizando la capa radio en clase A. Es posible cambiar la clase a B o C cambiando el código C# correspondiente. Pueden modificarse también parámetros de la comunicación como el Duty cycle, la tasa, hacer un ping periódico o la región en la que transmitimos. Para ejecutarla hay que cargar el fichero .project que se encuentra dentro de la ruta STM32Cube\_FW\_WL\_V1.0.0/ Projects/ NUCLEO-WL55JC/ Applications/ LoRaWAN/ LoRaWAN\_End\_Node. Esto cargará el proyecto en el IDE, donde puede compilarse (herramienta martillo) y ejecutarse en la placa dándole al play. Es sin duda la aplicación más interesante y ligada a lo que hemos visto anteriormente en los artículos de uso de LoRaWAN en salud conectada. En este caso no envía parámetros biológicos, sino que envía la temperatura y la batería del microprocesador, aunque el funcionamiento sería análogo. No ha podido probarse porque es necesario conectar el dispositivo a un Gateway para formar la red LoRaWAN y no se disponía de dicho dispositivo.

- LoRaWAN\_AT\_Slave:

Este ejemplo implementa las capas de LoRaWAN y LoRa para controlarlas mediante comandos AT [16] por una interfaz UART. Puede ejecutarse el .project en el IDE igual que antes en la ruta STM32Cube\_FW\_WL\_V1.0.0/ Projects/ NUCLEO-WL55JC/ Applications/ LoRaWAN/ LoRaWAN\_AT\_Slave. Otra opción es ejecutar el flujo en el STM32Monitor. Es una

manera de ejecutar igualmente la aplicación, pero con una interfaz visual más atractiva. En caso de utilizar el IDE es necesario conectarse por puerto serie a la placa para ver el intercambio de mensajes. Para esto hay varias aplicaciones, yo he usado Tera Term [17] en Windows, aunque en las aplicaciones finales que he podido utilizar no era necesario porque las he ejecutado a través del STM32CubeMonitor. Refiriéndonos concretamente a esta aplicación (LoRaWAN\_AT\_Slave) he tenido que lidiar con varios errores al importar el flujo al SMT32CubeMonitor que no me permitían ejecutar la aplicación. Tras conseguir subsanarlos pudimos probar esta aplicación, cuya configuración y resultados se presentan en el apartado V.

En último lugar, mencionar que también encontramos en STM32Cube\_FW\_WL\_V1.0.0/ Projects/ NUCLEO-WL55JC/ Demonstrations/ LocalNetwork los archivos binarios para poder probar esta demostración. Dicha demostración consiste en una red en estrella donde uno de los nodos finales funciona como concentrador y el otro como sensor. La aplicación permite tener catorce sensores (que serían catorce STM32WL55) conectados a un único concentrador. Este concentrador no hace las tareas de Gateway, es decir, no vamos a tener funcionalidades LoRaWAN porque el concentrador usado (recordemos que es un STM32WL55, que realmente es un nodo final) no permite el reenvío de paquetes por IP. Esta demostración solo pretende mostrar cómo se manejaría una red de sensores vista solo en la parte comprendida entre el Gateway y los dispositivos finales, es decir, se centra en la capa física LoRa. El estudio realizado con esta demo y los resultados obtenidos se presentan en el apartado VI.

Puesto que no contábamos con el hardware necesario para montar una red LoRaWAN completa hemos probado las dos últimas aplicaciones mencionadas (AT\_Slave y LocalNetwork), las cuales sí se pueden ejecutar con los dos dispositivos finales con los que contamos.

## V. LORAWAN AT SLAVE

### A. Configuración

Para utilizar esta aplicación son el STM32CubeMonitor se han seguido los pasos descritos en [18]. Para poder utilizar la aplicación es necesario realizar los siguientes pasos:

- En primer lugar, se conecta uno de los dispositivos por USB y se abre el STM32CubeProgrammer. Nos conectamos al dispositivo y lo reiniciamos para trabajar sin errores. Una vez reseteado lo desconectamos del STM32CubeProgrammer:

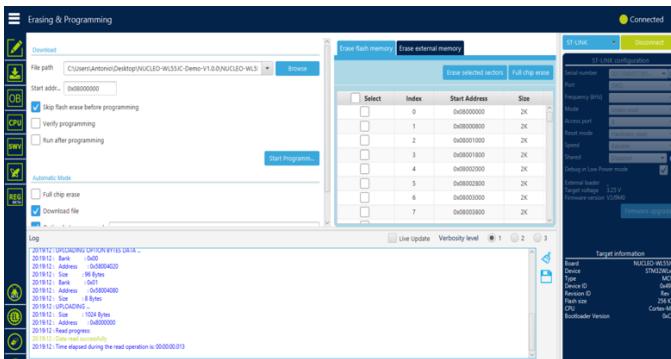


Fig. 10. Interfaz STM32CubeProgrammer

- Tras esto abrimos el .project de esta aplicación en el IDE y lo compilamos y ejecutamos en la placa:

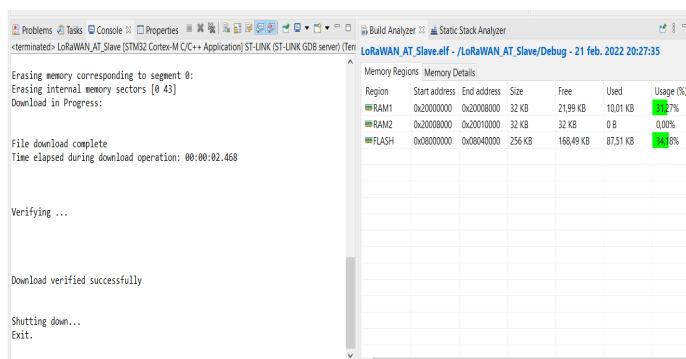


Fig. 11. Ejecutando programa en la placa con STM32CubeIDE

- Desconectamos la placa y conectamos la otra, realizando los dos pasos anteriores del mismo modo.
- Abrimos el STM32CubeMonitor e importamos el flujo que está en la misma ruta, pero en la carpeta STM32CubeMonitor (es un fichero .json). Si da error puede volver a cargarse y parece que se soluciona. Una vez cargado hay que conectar los dos dispositivos por USB al ordenador y en los flujos Serial Port A y Serial Port B debemos modificar los valores para poner uno COM3 y en otro COM4, que corresponden a los puertos serie de ambos dispositivos. Le damos a Deploy y ejecutamos la demo con el botón Dashboard. Abrimos el RF test LoRa (también se puede hacer la prueba con la FSK) y podemos ejecutar varios tests:

- RSSI, para ver la fuerza de la señal recibida.
- Transmitir una señal continua.
- Medir la tasa PER (packet error rate) en la transmisión entre los dos dispositivos.

## B. Resultados

Veamos algunos resultados obtenidos al ejecutar los diferentes tests:

- RSSI:

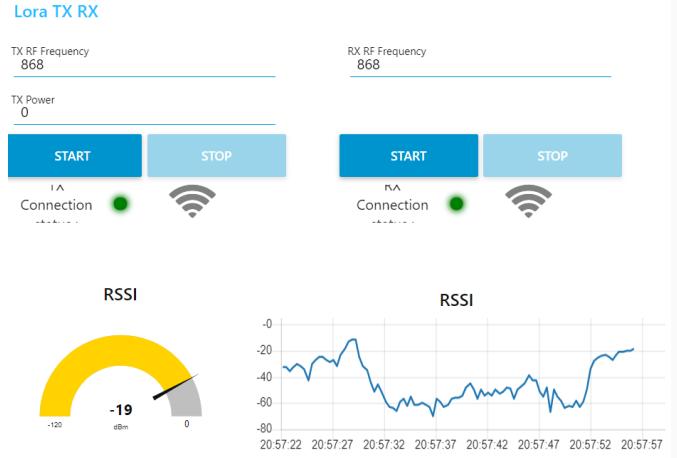


Fig. 12. Resultado en STM32CubeMonitor al ejecutar el test RSSI

Vemos como varía la fuerza de la señal transmitida a lo largo del tiempo. Para provocar estas variaciones se ha movido el dispositivo (dentro de la gran limitación de movilidad que supone tenerlos ambos conectados por USB al mismo ordenador). Al alejarlos o cubrirlos con la mano se obtienen valores de RSSI de unos -40 a -80 dBm, mientras que si los acercamos mucho la fuerza de la señal recibida aumenta hasta unos -10 dBm. Puede probarse a configurar una frecuencia distinta de emisión y recepción, comprobándose que en ese caso la señal recibida tiene muy poca fuerza (unos -90 dBm solo al cambiar de 868MHz a 869MHz).

- PER Test: Este test permite computar la packet error ratio (PER). Un paquete se declara como incorrecto si uno o más bits no coinciden. Se configura la prueba inicialmente con un ancho de banda de 125 kHz, un SF de 12 y una tasa de codificación de 4/6, transmitiendo 5 tramas con una potencia de 0 dBm.

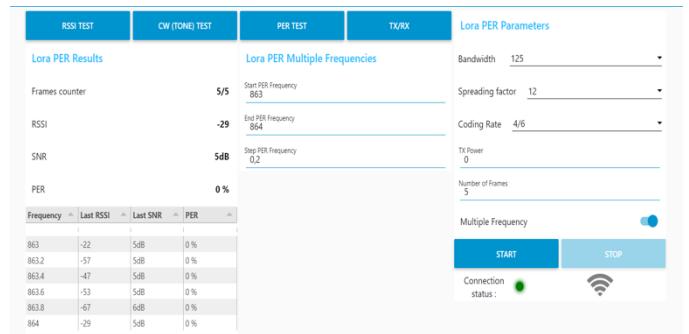


Fig. 13. Resultado en STM32CubeMonitor al ejecutar el test PER

Probamos a variar algunos parámetros. Por ejemplo, en esas mismas condiciones probamos con una potencia de transmisión de -9 dBm, la más baja disponible. La RSSI baja de forma considerable mientras la SNR permanece más o menos constante y siguen sin producirse errores.

Podemos probar también a transmitir con 0 dBm ajustando el SF a 5 y la tasa de codificación a 4/8. En esas condiciones:

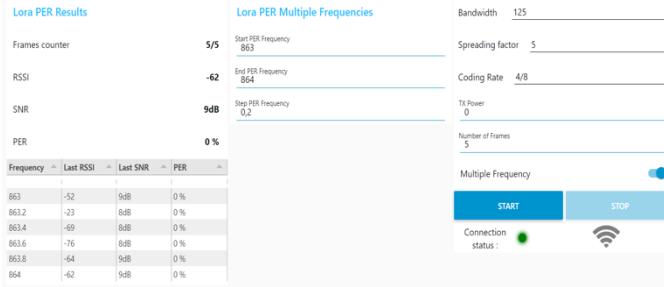


Fig. 14. Resultado en STM32CubeMonitor al ejecutar el test PER con otra configuración

Podemos comentar de la ejecución que es mucho más rápida, algo esperable al reducir el SF, ya que como sabemos al bajarlo aumenta la tasa de la comunicación. La SNR ha mejorado bastante y la RSSI se comporta igual: más alta cuando los situamos más cerca y más baja cuando los alejamos más. En ninguna de estas dos configuraciones (ni en ninguna de todas las probadas durante la realización de este trabajo) se ha conseguido perder algún paquete. Sería interesante probar de otra forma, pudiendo alejar los dispositivos a grandes distancias (recordemos que LoRa permite comunicaciones en un rango amplio). Esto se estudiará en la siguiente prueba.

El test que falta consiste en transmitir continuamente paquetes y puede verse in situ como ambos dispositivos encienden los LEDs de forma continua. También pueden verse los comandos AT intercambiados en el apartado AT CMD.

## VI. LOCALNETWORK

### A. Configuración

La otra aplicación probada es la demo del concentrador al que se conectan hasta catorce sensores. En este caso uno de los dispositivos hará de concentrador y el otro de sensor, enviando periódicamente datos.

Para poner en marcha esta aplicación hay que seguir los pasos descritos en [19]:

- En primer lugar, conectamos uno de los dispositivos al STM32CubeProgrammer y lo reiniciamos.

- En el propio STM32CubeProgrammer cargamos el archivo binario correspondiente al concentrador, el cual está en la ruta NUCLEO-WL55JC – Demo-V1.0.0/ NUCLEO-WL55JC-Demo-Concentrator-V1.0.0.bin:

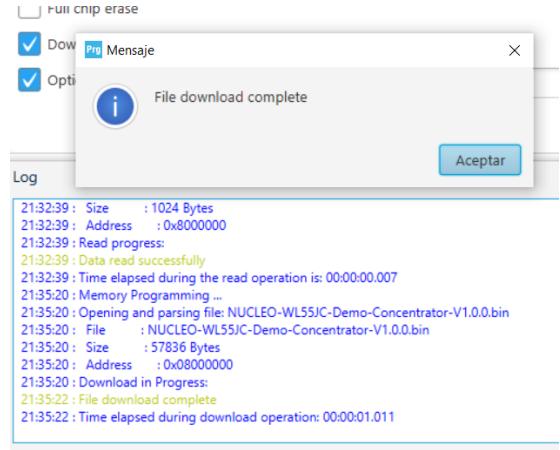


Fig. 15. Resultado en STM32CubeProgrammer la cargar el archivo binario del concentrador

- Desconectamos este dispositivo y conectamos el otro, realizando los mismos pasos, pero con cuidado de cargar el archivo binario del sensor y no el del concentrador.

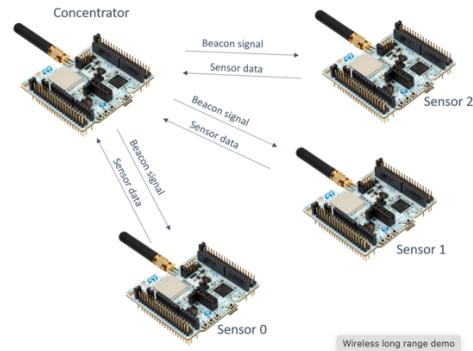


Fig. 16. Configuración de la demostración [20]

- Desconectamos el dispositivo sensor y conectamos el concentrador. Cargamos a continuación el modelo .json en el STM32CubeMonitor desde la ruta STM32Cube\_FW\_WL\_V1.0.0/ Projects/ NUCLEO-WL55JC/Demonstrations/LocalNetwork/LocalNetwork\_Concentrator/ SMT32CubeMonitor. En el apartado de configuración de puerto serie ponemos el del dispositivo conectado, el concentrador. Clicamos en Deploy y abrimos la dashboard.

- Al empezar la demo se pide seleccionar la región, en nuestro caso será la Unión Europea, a una frecuencia de 869.525MHz. En este punto el concentrador se queda esperando hasta que se conecte un sensor. Cada 16 segundos el concentrador emite un beacon para que los sensores se conecten. El siguiente time slot posterior a

la beacon es para sincronización y los 14 time slots restantes son para que los sensores le envíen información.

- Con un cargador de teléfono conectamos el sensor a la corriente en algún punto y vemos como tras unos segundos se establece la conexión correctamente. En ese instante empiezan a aparecer los comandos AT e información relativa a los mensajes recibidos, perdidos, la RSSI, la PER, la SNR y la temperatura que envían los sensores. En este caso envían la temperatura del microprocesador como dato, aunque es escalable a una aplicación biomédica cambiando la temperatura del controlador por otra magnitud.
- Los parámetros de la conexión LoRa son: SF = 11, ancho de banda de 125kHz, tasa de codificación de 4/5 y un tamaño de mensaje de 4 bytes.

## B. Resultados

Este ejemplo nos da más juego para probar diferentes escenarios, ya que no hay que tener ambos dispositivos conectados al ordenador. En mi caso probé varias situaciones para ver si enviaba bien la temperatura y para comprobar qué tal se comportaba la comunicación para distancias grandes y con obstáculos, tanto en interiores como en exteriores.

En primer lugar, conecté el sensor a unos 50 cm del concentrador. La temperatura medida era de unos 18°C. En segundo lugar, coloqué el sensor cerca de la estufa a una distancia prudente, viendo como la temperatura aumentaba hasta unos 23°C. En tercer lugar, se aumentó la potencia de la estufa, viendo como el sensor medía correctamente la temperatura al aumentar esta hasta 26°C. El siguiente experimento consistió en llevar el sensor hasta la habitación más alejada de la casa, cerrando las puertas de ambas habitaciones. En ese caso la temperatura fue bajando de nuevo hasta el valor de temperatura ambiente y vimos como la RSSI y la SNR disminuyeron, ya que la distancia pasó de ser de varios centímetros hasta unos 10 metros. Una nueva prueba consistió en llevar el concentrador a la azotea del edificio donde vivo, manteniendo el sensor en la habitación más alejada, a unos 12 metros en línea recta, pero con dos techos de por medio. En este caso movimos el concentrador porque va conectado al portátil, no pudiendo mover el sensor al no contar con toma de corriente en la azotea. Vemos aquí como la RSSI y la SNR siguen bajando. El último experimento consistió en dejar el sensor en la ventana de la casa y llevar el concentrador conectado al portátil en el coche, mirando en distintas ubicaciones para ver hasta qué distancia podíamos cubrir.

Se presenta a continuación el recorrido realizado en exteriores en coche para probar la conectividad en función de la distancia:



Fig. 17. Plano del experimento realizado en exteriores.

Cada punto rojo corresponde a una de las paradas para comprobar que la comunicación seguía estable. Se marcan la ubicación del sensor fijo y la zona de pérdida de conexión. Se ha comprobado en Google Maps que la distancia entre ambos puntos es de aproximadamente 800 metros. En este punto se comprobó el estado de la comunicación en el STM32CubeMonitor, obteniendo los siguientes resultados, los cuales abarcan desde la conexión inicial con ambos dispositivos cercanos hasta la prueba final en exteriores:

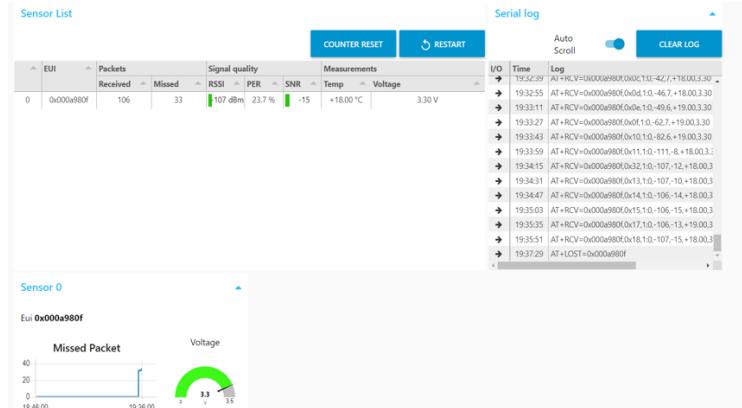


Fig. 18. Resultados del estudio en interiores y exteriores en STM32CubeMonitor

Se aprecia como durante todo el estudio se pierden 33 paquetes y 106 se reciben correctamente. Podemos mencionar que se comprobó que dichos paquetes se pierden en las proximidades de la zona de desconexión, siendo el resto de la comunicación exitosa. La PER es del 23.7%, ya que de los 139 paquetes totales se pierden 33. Los valores de RSSI y SNR que aparecen son los de la última comunicación. También puede verse el log con todos los mensajes intercambiados entre el concentrador y el sensor.

Podemos ver también las fluctuaciones de temperatura a lo largo del tiempo, teniendo en cuenta que la zona de pico es cuando acercamos el sensor a la estufa y el resto del tiempo se equilibra en torno a 20°C, que era la temperatura ambiente en el momento de la prueba.

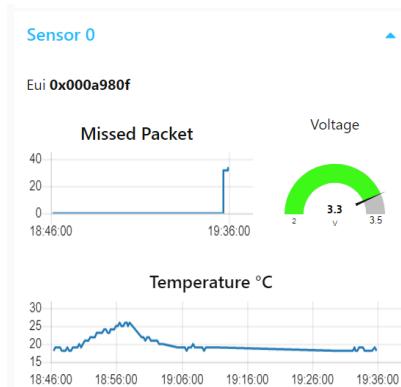


Fig. 19. Resultados del estudio en interiores y exteriores en STM32CubeMonitor

Puesto que los valores de RSSI y SNR que ofrece el STM32CubeMonitor son solamente los de la última comunicación se ha escrito un pequeño script en MATLAB que analice el log de la comunicación para poder generar unos gráficos de interés, los cuales presentamos a continuación:

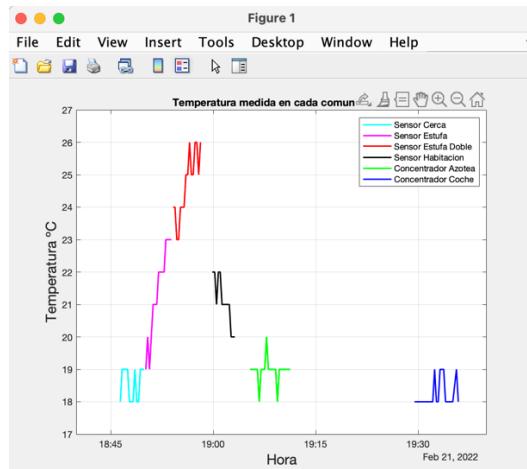


Fig. 20. Resultados del estudio en interiores y exteriores: temperatura

Vemos en primer lugar las variaciones de temperatura para cada experimento. El comportamiento es el esperado, tal y como hemos mencionado anteriormente.

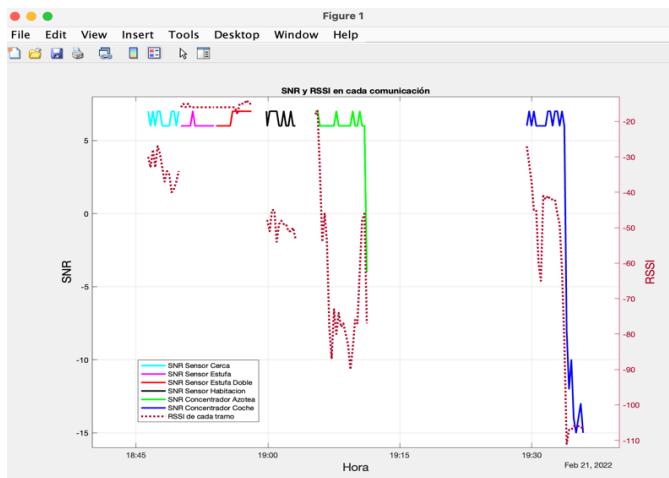


Fig. 21. Resultados del estudio en interiores y exteriores: SNR y RSSI

En esta figura se presenta la SNR y la RSSI medida en cada uno de los experimentos. Estudiando la gráfica vemos que en los tres primeros (donde el concentrador y el sensor estaban muy cercanos), tanto la SNR como la RSSI son bastante buenas.

Al alejar el sensor y llevarlo a la habitación más alejada vemos como la SNR tampoco parece alterarse demasiado, a pesar de pasar de estar a pocos centímetros a estar a unos 10 metros. La RSSI sí que baja en este cuarto escenario, ya que vemos que pasó de estar en torno a -20 dBm a estar en torno a -50 dBm. Esto es esperable por la nueva distancia que ha de superar la comunicación, habiendo obstáculos como puertas o paredes de por medio.

Al llevar ahora el concentrador a la azotea vemos como acaba bajando bastante tanto la RSSI como la SNR. La RSSI llega a bajar hasta los -80 dBm y la SNR hasta unos -4 dB. Esto es normal, ya que ahora nos situamos a más distancia, con uno de los nodos en interior y otro en exterior y con dos techos de por medio.

Por último, al hacer el recorrido en coche vemos como la SNR y la RSSI van disminuyendo bastante a medida que pasa el tiempo, ya que nos íbamos alejando progresivamente. En el punto donde se pierde la comunicación tenemos unos -15 dB de SNR y unos -110 dBm de potencia de señal recibida.

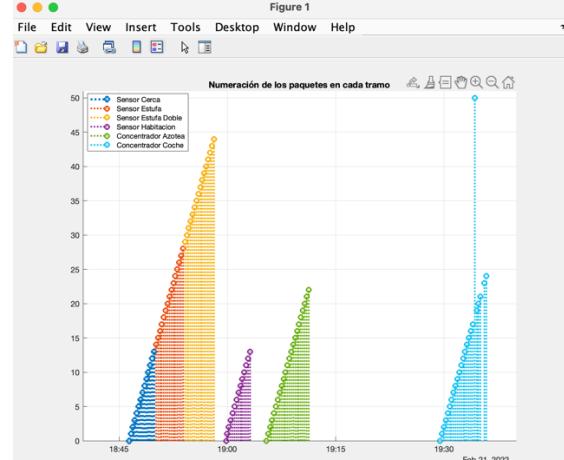


Fig. 22. Resultados del estudio en interiores y exteriores: Packet number

Esta última gráfica permite comprobar que se reciben bien los paquetes. En esta aplicación los paquetes se numeran en formato hexadecimal de forma creciente y consecutiva. En los tres primeros experimentos vemos como funciona como debe, ya que el número del paquete recibido es siempre uno mayor al anterior. El cuarto experimento empieza de cero porque se desconectó el sensor de la corriente para conectarlo en otra habitación. Lo mismo ocurre en el quinto experimento correspondiente a la azotea. Donde sí vemos problemas es en el experimento en exteriores, ya que vemos que llega un punto donde un paquete pasa de tener identificador 17 a identificador

50. Además, es curioso que el siguiente tiene identificador 19 y poco después hay un paquete que se pierde. Este comportamiento es bastante singular y complejo de analizar, ya que no lo asociaría a que se haya perdido 33 paquetes (uno perdido y los 32 de diferencia entre el 17 y el 50). Pienso que no se pierden porque si así fuera debería haber 32 huecos de mensajes no recibidos entre el mensaje 17 y el 50, sin embargo, lo que ocurre es que llegan consecutivamente como deben, pero con unos identificadores muy dispares. No consigo identificar el por qué de este comportamiento, pero quizás se deba al mal estado de la conexión en ese punto del recorrido. En el caso del mensaje perdido no hay confusiones porque se aprecia claramente el hueco del mensaje que falta.

En definitiva, todas estas pruebas nos han servido para estudiar en detalle las aplicaciones de LoRa en estos dispositivos STM32WL55. Ha sido realmente interesante poder hacer tests variando parámetros de la modulación y poder montar una pequeña red punto a punto con un sensor y un concentrador, viendo cómo varía la temperatura enviada en distintos escenarios y sobre todo comprobando el estado de la conexión en diferentes escenarios, tanto en interiores como en exteriores.

La conexión más larga ha sido de unos 700 metros. Es un rango considerable para unos dispositivos como estos, aunque lejos de los 2 km que consigue ofrecer LoRa. Podría mejorarse este desempeño ubicando el nodo fijo en un punto más alto e intentando movernos a zonas con menos edificios alrededor. También pienso que la presencia el río puede tener algo que ver, aun así, ha sido muy interesante poder comprobar in situ las bondades de estos dispositivos.

## VII. LÍNEAS FUTURAS

Como posible línea futura de trabajo se propone el empleo de un dispositivo Gateway para poder montar una red LoRaWAN completa. El dispositivo necesario es el P-NUCLEO-LRWAN2, el cual cuenta con un puerto ethernet para conectarse a internet.

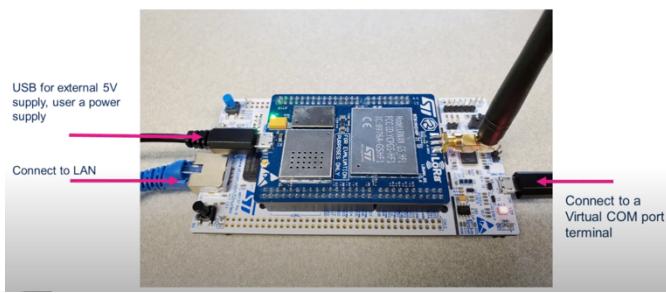


Fig. 23. Gateway de STM [12]

Para poder trabajar en esta línea futura recomiendo el tutorial compuesto de 6 vídeos que puede encontrarse en [12]. Es realmente interesante porque explican paso a paso como configurar el Gateway y cómo crear la red en The Things

Networks. Al ver los vídeos se deduce que montar la red con The Things Networks parece bastante sencillo e intuitivo, siendo quizás la parte más compleja la configuración del dispositivo final, ya que hay que modificar algunos parámetros de la conexión en el propio código C#.

También proponen y describen cómo usar un servidor de aplicación llamado Cayenne myDevices, el cual permite ver los datos recibidos de los sensores con una interfaz gráfica muy amable e intuitiva.

Otro complemento que se puede añadir es un conjunto de sensores, en este caso de humedad, temperatura y presión. Estos sensores van en la placa X-NUCLEO-IKS01A3, la cual se puede conectar a los conectores Morpho del STM32WL55 que hemos usado.

Para finalizar, conectan al STM32WL55 un dispositivo llamado X-NUCLEO-LPM01A que permite programar el voltaje que se suministra al dispositivo. Emplean a su vez el software SMT32CubeMonitor Power para medir el consumo de potencia de la placa, describiendo cómo hacer el montaje y cómo configurar todos los programas:

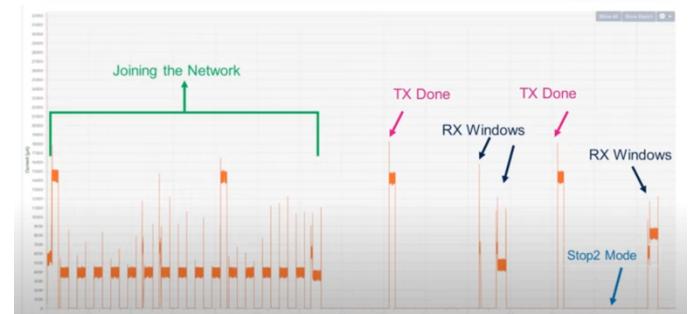


Fig. 24. Control de consumo de energía en STM32CubeMonitor Pwr[12]

Vemos en la imagen anterior la monitorización de consumo de energía que presentan en el tutorial que menciono. Destaco esta imagen porque se aprecia muy bien como tras una transmisión aparecen las dos ventanas de recepción, por lo que podemos deducir que se trata de una transmisión LoRa con el dispositivo configurado como clase A. También hubiera sido interesante usar este software para analizar el consumo de energía en las aplicaciones que he probado, algo que no he podido llevar a cabo al no reconocer el software los dos STM32WL55 que he empleado.

También se propone como trabajo futuro probar los múltiples ejemplos que proporciona el fabricante, ya que este trabajo es solo una primera aproximación y queda bastante por explorar.

## REFERENCES

- [1] Lora Alliance. (2015). *LoRaWAN: What is it?* Disponible en: <https://lora-alliance.org/> [Consultado: 20/11/2021]
- [2] SEMTECH. (2021). LoRaWAN Standard Disponible en: <https://www.semtech.com/lora/lorawan-standard> [Consultado: 22/11/2021]

- [3] Boloix Tortosa, R. (2021). *Long Range Interfaces for IoT: LPWAN*. Universidad de Sevilla.
- [4] Alliot Technologies. (2018). What is LoRaWAN? Disponible en: <https://www.alliot.co.uk/2018/07/what-is-lorawan/> [Consultado: 22/11/2021]
- [5] Sanchez-Iborra R, Cano M-D. (2016). State of the Art in LP-WAN Solutions for Industrial IoT Services. *Sensors*, 16(5). DOI: 10.3390/s16050708
- [6] M. T. Buyukakkaslar, M. A. Erturk, M. A. Aydin and L. Vollero, "LoRaWAN as an e-Health Communication Technology," 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC), 2017, pp. 310-313, doi: 10.1109/COMPSAC.2017.162.
- [7] A. Mdhaffar, T. Chaari, K. Larbi, M. Jmaiel and B. Freisleben, "IoT-based health monitoring via LoRaWAN," IEEE EUROCON 2017 -17th International Conference on Smart Technologies, 2017, pp. 519-524, doi: 10.1109/EUROCON.2017.8011165.
- [8] J. M. González-García and C. Estevez, "Smart Hospital LoRaWAN-based Backup Network Design that Monitors Critical Emergency and Standby Resources," 2019 IEEE International Conference on E-health Networking, Application & Services (HealthCom), 2019, pp. 1-6, doi: 10.1109/HealthCom46333.2019.9009604.
- [9] Dimitrievski, A.; Filiposka, S.; Melero, F.J.; Zdravevski, E.; Lameski, P.; Pires, I.M.; Garcia, N.M.; Lousado, J.P.; Trajkovik, V. Rural Healthcare IoT Architecture Based on Low-Energy LoRa. *Int. J. Environ. Res. Public Health* 2021, 18, 7660. <https://doi.org/10.3390/ijerph18147660>
- [10] Arm Limited. 2022. Nucleo-WL55JC. Disponible en: <https://www.mouser.es/new/stmicroelectronics/stm-stm32wl-nucleo-64/> [Consultado 20-02-2022].
- [11] Mouser Electronics. 2020. STMicroelectronics STM32W Nucleo-64 Board (NUCLEO-WL55JC).
- [12] Tijerina, J. 2021. LoRaWAN with SMT32 Getting Started. [YouTube]. Disponible en: <https://www.youtube.com/watch?v=uYULivHQoNI> [Consultado 20-02-2022].
- [13] Mouser Electronics. 2021. STM32WL55. Disponible en: <https://hu.mouser.com/c/?q=STM32WL55> [Consultado 21-02-2022].
- [14] STMicroelectronics. 2021. STM32CubeWL Nucleo firmware. Disponible en: [https://www.st.com/resource/en/user\\_manual/um2786-stm32cubewl-nucleo-demonstration-firmware-stmicroelectronics.pdf](https://www.st.com/resource/en/user_manual/um2786-stm32cubewl-nucleo-demonstration-firmware-stmicroelectronics.pdf) [Consultado 21-02-2022].
- [15] STMicroelectronics. 2022. STM32 Nucleo-64 development board with STM32WL55JCI MCU, SMPS, supports Arduino and morpho connectivity. Disponible en: <https://www.st.com/en/evaluation-tools/nucleo-wl55jc.html> [Consultado 21-02-2022].
- [16] Wikipedia. 2021. Conjunto de comandos Hayes. Disponible en: [https://es.wikipedia.org/wiki/Conjunto\\_de\\_comandos\\_Hayes](https://es.wikipedia.org/wiki/Conjunto_de_comandos_Hayes) [Consultado 21-02-22].
- [17] Tera Term Project. 2021. Tera Term Home Page. Disponible en: <https://ttssh2.osdn.jp/index.html.en> Consultado [21-02-22].
- [18] STMicroelectronics. 2022. Wireless Long Range RF Test. Disponible en:[https://wiki.st.com/stm32mcu/wiki/STM32CubeMonitor:Wireless\\_Lo](https://wiki.st.com/stm32mcu/wiki/STM32CubeMonitor:Wireless_Lo)ng\_Range\_RF\_Test [Consultado 20-02-22].
- [19] STMicroelectronics. 2022. How to start Wireless Long Range demo. Disponible en: [https://wiki.st.com/stm32mcu/wiki/STM32CubeMonitor:Wireless\\_Lo](https://wiki.st.com/stm32mcu/wiki/STM32CubeMonitor:Wireless_Lo)ng\_Range\_RF\_Test [Consultado 20-02-22].
- [20] STMicroelectronics. 2021. STM32WL55JC boards. Disponible en: [https://www.st.com/content/ccc/resource/training/technical/product\\_training/group1/dd/f1/d0/1a/d0/da/42/24/STM32WL-Ecosystem-STM32WL\\_Boards\\_BOARD/files/STM32WL-Ecosystem-STM32WL\\_Boards\\_BOARD.pdf\\_jcr\\_content/translations/en.STM32WL\\_Ecosystem-STM32WL\\_Boards\\_BOARD.pdf](https://www.st.com/content/ccc/resource/training/technical/product_training/group1/dd/f1/d0/1a/d0/da/42/24/STM32WL-Ecosystem-STM32WL_Boards_BOARD/files/STM32WL-Ecosystem-STM32WL_Boards_BOARD.pdf_jcr_content/translations/en.STM32WL_Ecosystem-STM32WL_Boards_BOARD.pdf) [Consultado 20-02-2022].