Proyecto Fin de Carrera
Ingeniería Electrónica, Robótica y Mecatrónica

Diseño de un HUB Wireless M-Bus-NbIoT.

Autor: Alberto Ávila Soriano
Tutor: Ramón González Carvajal

Dpto. Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Proyecto Fin de Carrera
Ingeniería Robótica, Electrónica y Mecatrónica

Diseño de un HUB Wireless M-Bus-NbIoT.

Autor:
Alberto Ávila Soriano

Tutor:
Ramón González Carvajal
Profesor Catedrático

Dpto. de Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2021

Proyecto Fin de Carrera: Diseño de un HUB Wireless M-Bus-NbIoT.

Autor: Alberto Ávila Soriano

Tutor: Ramón González Carvajal

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

A mi familia

A mis maestros

Agradecimientos

Empiezo con mi profesor, tutor de este trabajo de fin de grado y la persona que me ha dado la oportunidad de hacer un TFG interesante y enriquecedor, Ramón. Muchas gracias de todo corazón por tu enseñanza en sentido más amplio de la palabra. Tus clases me han ayudado a ser un mejor ingeniero pero más importante, una mejor persona. Definitivamente ha sido una pieza clave durante mi periodo académico que ha hecho que vea el mundo que hay afuera de una forma más objetiva y clara. Simplemente gracias por todo.

Agradecimientos a mis padres por estar ahí siempre que lo he necesitado y ser un apoyo fundamental en mi vida, a mi hermano por enseñarme a ser una mejor persona y al resto de mi familia por darme una razón para trabajar cada día. A mis niños del colegio del Carmen por ver en ellos la otra cara de la moneda y ver que se puede tener hermanos sin ser de tu misma sangre.

A Juanlu y Sergio por ser la razón por la que a Sevilla la llamo hogar. A mis amigos de Erasmus pero en especial a Mario y Esteban por enseñarme que la amistad es un diamante, y los diamantes son para siempre.

Y por último pero no menos importante, a Pablo. Mi compañero de fatigas y de alegrías. Tanto a nivel académico como personal, y por seguir siendo mi compañero en la vida y en esta experiencia que recorreremos juntos. Muchas gracias por estar a mi lado.

Resumen

El agua es un recurso indispensable para la vida en nuestro planeta y para la gran mayoría de las actividades económicas que se realizan. No es ampliable por mera voluntad del ser humano y para ciertas actividades que requieren agua dulce y limpia, es tremendamente escasa. Por este último motivo es un recurso susceptible de ser contaminado fácilmente y de ser usado repetidas veces. Esta susceptibilidad del recurso, sumado a la expansión demográfica, la contaminación y el envejecimiento de las infraestructuras hacen del agua un recurso valioso, delicado y del que tenemos la necesidad fundamental de mantenerlo de forma sostenible.

Los sistemas convencionales de transporte y distribución del agua limpia son poco eficaces a la hora de tomar datos, ya que dicha extracción se puede obtener sólo manualmente y consumos generalmente domésticos, y a la hora de reparación de alguna avería ya que son relativamente difíciles de encontrar y su reparación es costosa.

El objetivo de este proyecto será el desarrollo e implementación de un dispositivo inteligente de bajo consumo para minimizar el mantenimiento. Económico y fiable para que sea atractivo a los consumidores de dicho producto, con el objetivo de conseguir un producto acorde con la urgente necesidad del sector de tratar el agua de forma sostenible.

Abstract

Water is an essential resource for life on our planet and for most of the economical activities we do every day. It is not expandable by human will and, some activities are required of fresh and clean water in order to work, and this resource is incredibly scarce. For this reasons, clear water is up to be polluted and used and treated multiple times. The susceptibility of this resource added to the demographic growth, the pollution and the constant aging of the infrastructures make the water a delicate and valuable resourse which is needed of a sostainable way of processing.

The habitual ways of transport and distribution of clean water are not so effective in the data analysis, basically because the data collection just can be taken manually and usually, just the domestic consumption, and at the same time to locate and repair a failure in the system due to the difficulty of finding and the cost derivate to the repair.

The aim of this project is the develop and the implementation of a low-power intelligent devide to minimize the maintenance. Economic and reliable to catch the consumer's interest with the main objective of trating water on a sostenible way.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Notación

A^*	Conjugado
c.t.p.	En casi todos los puntos
c.q.d.	Como queríamos demostrar
■	Como queríamos demostrar
e.o.c.	En cualquier otro caso
e	número e
IRe	Parte real
IIm	Parte imaginaria
sen	Función seno
tg	Función tangente
arctg	Función arco tangente
sen	Función seno
$\sin^x y$	Función seno de x elevado a y
$\cos^x y$	Función coseno de x elevado a y
Sa	Función sampling
sgn	Función signo
rect	Función rectángulo
Sinc	Función sinc
∂y ∂x	Derivada parcial de y respecto
$x_\circ$	Notación de grado, x grados.
Pr(A)	Probabilidad del suceso A
SNR	Signal-to-noise ratio

MSE	Minimum square error
:	Tal que
<	Menor o igual
>	Mayor o igual
\	Backslash
⇔	Si y sólo si

1 INTRODUCCIÓN

La reutilización del agua limpia de forma sostenible se ha vuelto un requisito fundamental para los gobiernos de todo el planeta. El agua se usa en la mayoría de actividades de la vida cotidiana, a parte de ser de vital importancia en las actividades agrarias y ganaderas. Sin mencionar la necesidad de agua limpia para el desarrollo humano que ya, es una realidad que es una escasez y muchas personas no tienen acceso a ella [1].

En España, se redactó un plan y una normativa[2] para aumentar la renovabilidad del agua. Pero por falta de recursos sólo el 10,4% de las aguas en nuestro país se renueva de forma segura para el medio ambiente. Se ha ido produciendo un descenso de los fondos invertidos en temas relacionados al agua como infraestructuras, que ya de por sí están anticuadas y son poco eficientes.

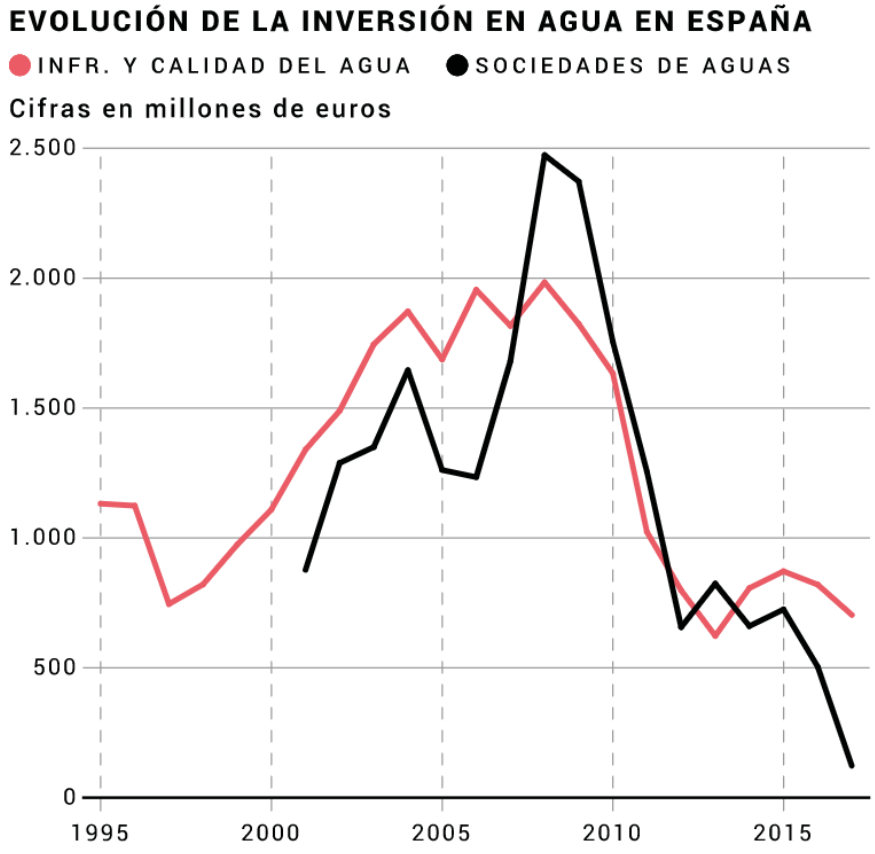


Figura 1.1- Descenso de los fondos destinados al agua en millones de Euros.

Este descenso de fondos sumado a la importancia que tiene el agua y las épocas de sequía que llevamos sufriendo en algunas zonas de nuestro país durante años, hacen del agua un recurso de vital importancia que tiene que ser tratado de forma consecuente. El sector público en España está en la cola de inversión de agua, por lo que le tocaría al sector privado tomar el relevo y proponer soluciones viables y económicas para el tratamiento de aguas.

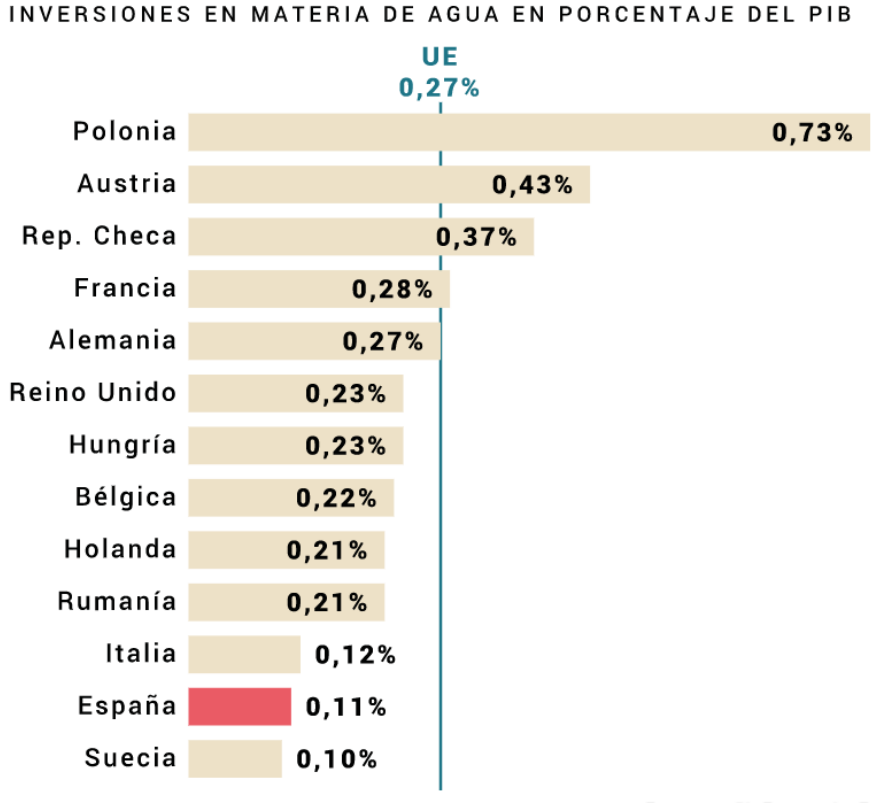


Figura 1.2- Porcentaje del PIB de cada país destinado al agua.

A día de hoy, se observan numerosas e importantes carencias y se hace notar la falta de sistemas de gestión integrados que abarquen todas las áreas implicadas. Dentro de estas carencias se pueden hacer notar por ejemplo la falta de una toma de datos global. Los datos que se reciben actualmente sólo se obtienen de forma parcial, no global. Y dentro de esta toma de datos parcial solo conocemos el agua destinada al consumo doméstico y ni siquiera de forma telemática e informatizada.

Después los datos de la red de distribución y los pocos datos que se obtienen de las redes domésticas no se integran, de forma que las aproximaciones obtenidas del consumo no dejan de ser una aproximación alejada de la realidad obtenida de una serie de datos obtenidos de una forma no sistemática y/o centralizada. Por lo que se hace notar la falta de un sistema tecnológicamente viable, con costes asumibles por las operadoras y que represente un coste energético medioambientalmente equivalente al ahorro de agua perseguido.

Existen varias problemáticas en el tema del agua. La tecnología no es extrapolable a otros campos como puede ser el eléctrico, ya que las condiciones son diferentes y se necesitan de sensores, actuadores y transceptores que sean de ultra-bajo consumo capaces de operar extrayendo energía del medio en el que se encuentran o con baterías de muy larga duración para evitar un mantenimiento continuo, sumado a que tiene que ser de bajo coste para favorecer su despliegue masivo.

Es notable la necesidad de un sistema de tele-lectura que sea viable económicamente, que tenga un buen equilibrio de costes y beneficios y que tenga un coste energético que, en términos de cuidado del medio ambiente, no supere el ahorro de agua perseguido. La combinación de estos factores provoca que a día de hoy, no se realice una gestión eficiente del agua.

No obstante, la tele-lectura de contadores o telecontadores es ya una realidad tecnológicamente viable. A continuación, describiremos las diversas opciones que existen, sus pros y contras, haciendo hincapié en el protocolo de envio de datos inalambrico Wireless M-Bus. El motivo por el que no se han implantado de forma masiva tiene mucho que ver con la regulación actual (que todavía no obliga) y con el modelo de negocio de las empresas de aguas, especialmente las concesionarias, que no siempre fomenta la inversión en tecnología, primando la inversión en infraestructuras.

1.1 Motivación.

La motivación para realizar este proyecto reside en la utilidad del mismo. El agua es un recurso muy valioso para toda forma de vida y para la mayoría de actividades humanas, como dijo Leonardo Da Vinci “El agua es la fuerza motriz de toda la naturaleza”. Es prepcupante el crecimiento demográfico de cara a los recursos naturales, es una realidad que mucha gente carece de recursos básicos como la comida o el agua.

Por ese, entre otros motivos, este proyecto está cumpliendo una labor social. Está ayudando a mejorar la calidad de vida del mundo ya que con un mejor aprovechamiento del agua, podremos emplearla para otras actividades como la agricultura, generando, por ejemplo, más alimentos y ayudando a un crecimiento demográfico sostenible. El objetivo de este proyecto es conseguir un dispositivo robusto, fiable, económico y de bajo consumo para sustituir el mantenimiento constante por un mantenimiento puntual cuando se requiera.

También hay que hablar del interés económico del proyecto, ya este supone una mejora significativa en la calidad de las redes de transporte y distribución de agua en España a un precio mucho menor de lo que supondría la renovación total de dichas redes o el costoso mantenimiento actual de las mismas.

Además, que gracias a la tele-lectura de contadores con dispositivos de bajo consumo y en tiempo real, se pueden implantar sistemas en las redes de distribución de agua de las ciudades permitiendo un control mucho más exhaustivo del uso del agua, permitiendo así un ahorro significativo del agua como recurso natural, y haciendo mucho más eficientes las redes de distribución ya implantadas.

En conclusión, las múltiples implicaciones sociales y la mejora del ahorro y eficiencia de las redes de distribución no solo de España, si no de todo el mundo, es la motivación más grande que podría tener para implicarme y hacerlo con ilusión. La mejora de la calidad de vida es algo que el ser humano ha buscado siempre y es lo que impulsa el crecimiento económico. Y el hecho de poder brindar oportunidades para el uso del agua de forma más “desahogada” y controlada actúan de sobra como motivación válida para este proyecto.

1.2 Objetivos, alcance y requisitos.

El objetivo de este trabajo de fin de grado es la creación de un hub Wireless M-Bus-NbIoT, que concentre los datos recogidos por los diferentes contadores inalabricos que envían datos por el protocolo Wireless M-Bus, los concentre y los envíe en una banda más ancha por NbIoT.

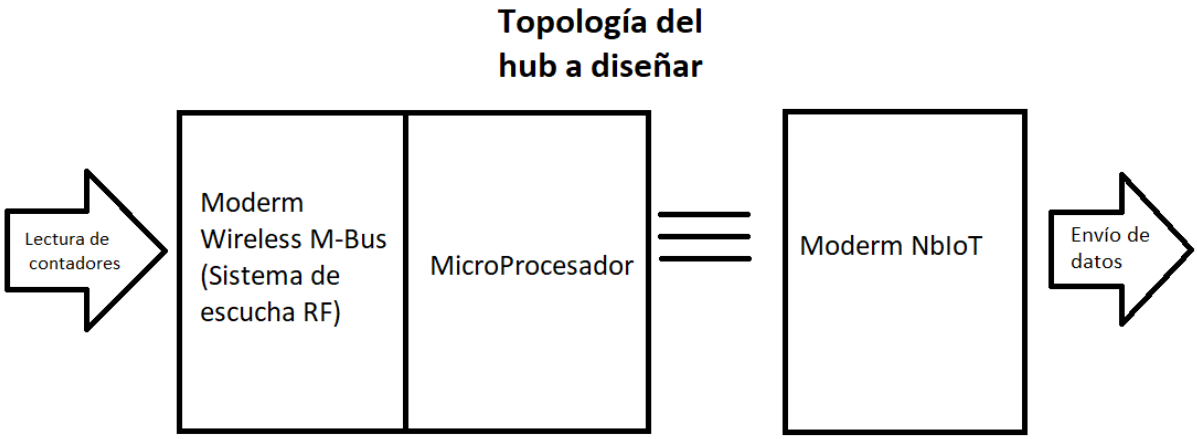


Figura 1.3- Topología del hub a diseñar.

El flujo de los datos va de la siguiente forma. Los datos enviados por el/los contadores son recibidos por el sistema de radio frecuencia del moderm Wireless M-Bus, el cual recibirá a través de comandos AT las órdenes, tramas y peticiones de respuesta enviadas por los diferentes contadores que transmiten en un rango de distancia relativamente bajo.

El microprocesador interpretará dichos datos y órdenes y tranformará la información de forma entendible para el moderm NbIoT, que recibirá la información a través de la UART y la procesará también, con comandos AT con el fin de enviarlas a un servidor, ya que NbIoT recorre un rango mucho mayor que Wireless M-Bus.

Por lo que el objetivo principal es la creación de este hub para poder aprovechar las ventajas tele-lectura por parte de los contadores Wireless M-Bus y su capacidad de concentrar la información de varios contadores sin necesidad de suscribciones a un operador, y la posibilidad de NbIoT de enviar los datos recibidos por UART a una distancia mucho mayor, posibilitando la opción de subir los datos a un servidor para su visualización e interpretación más sencilla.

El alcance de un proyecto como el que se está realizando radica la posibilidad de crear un sistema con ciertas características que permita un ahorro del agua como recurso a natural a gran escala favoreciendo su eficiencia y permitiendo usarla en diferentes ámbitos de la vida cotidiana. En este proyecto de fin de grado se pretende integrar el hub del que hemos hablado en un sistema de tele-lectura.

Algunas características de estos sistemas para que sea viable medioambientalmente hablando y que el ahorro de agua obtenido sea mayor que el consumo y esfuerzo del uso de estos sistemas se pueden ver reflejados a continuación.

Será necesario la creación de dispositivos con recursos limitados tanto de energía, memoria y capacidad de procesamiento. Ya que dichos nodos dependen de una batería, y para evitar coste de mantenimiento frecuente, tendrán que ser de bajo consumo, Lo cual por otra parte limita la capacidad de procesamiento.

Tambíen se necesitará un coste unitario bajo del nodo, ya que para su despliegue a gran escala y en las redes de distribución de agua se necesitan diversos y variados puntos de medida que irá

proporcional al tamaño de la red. Además que también habría que tener en cuenta los costes de despliegue y mantenimiento.

Se tendría que cumplir que los nodos estén preparados para operar en condiciones ambientales difíciles para poder operar en cualquier red de distribución de agua del planeta, ofreciendo así versatilidad para su implantación en diferentes puntos. Por esto mencionado anteriormente también será necesario cierto margen en las topologías y poniendo a disposición diferentes modos de despliegue. En conclusión, se necesita un sistema robusto, fiable y fácilmente integrados a alto nivel.

La recogida masiva de datos procedentes de los lectores, individuales, se complementa con las medidas agregadas procedentes de los lectores de consumo sectorial. La redundancia de estos datos podría conllevar problemas, por lo que será necesaria una integración adecuada de estos datos que antes de ser un problema, dirija a una gestión correcta del sistema y un mantenimiento de las redes adecuado.

Buscamos que los sistemas sean fiables y seguros. Por su implantación en los diferentes medios, los sistemas estarán expuestos a canales interferentes debido a la presencia de otros canales o de obstáculos físicos, por lo que hay que tener en cuenta y diseñar un sistema que además de ser capaz de operar en estas condiciones, nos ofrezca una seguridad en las transmisiones frente a ataques, ya sean pasivos o activos.

Los sistemas de comunicaciones deben establecer las conexiones y mantener la conectividad de la red de una manera autónoma, de forma que, en una red como esta, con un número elevado de nodos, la información siga su flujo establecido y llegue al punto final sin incidencias. Para esto el protocolo sobre el que vamos a trabajar, Wireless M-Bus, ofrece una gran ventaja al tener la posibilidad de concentrar la información de varios nodos de forma sencilla.

Los sistemas de comunicaciones deberían permitir que los contadores fueran remotamente accesibles. Para ello se disponen plataformas de control que actúan como pasarelas, para su integración en arquitecturas IP (Internet Protocol), aunque ambién se han propuesto arquitecturas en que los propios nodos tienen conectividad IP.

Los requisitos de este proyecto radican, al final, en los requisitos a cumplir por un sistema de tele-lectura, los cuales radican en la resolución de ciertos problemas comentados a continuación.

- *Necesitamos un diseño eficiente en recursos y consumo para suplir esta carencia de recursos y fuente de alimentación. El uso de la energía disponible de manera inteligente permite alargar la vida útil de estos nodos de 8 a 10 años de forma que el consumo de energía de los contadores no influya con el de tele-lectura.*
- *Para suplir las diversas topologías y condiciones ambientales, necesitamos una arquitectura de red escalable y fácilmente integrable.*
- *Se necesita una mejora de los mecanismos de rutado y alcande de radio frecuencia de los dispositivos para suplir la cobertura de los nodos.*
- *Para la redundancia de los datos comentada anteriormente se dispondrá de una fusión de datos y un procesamiento localizado de los mismos.*
- *Debe tener tolerancia a fallos y ser fiable. Se precisa de herramientas de verificación y corrección de los datos en cada capa del protocolo a construir, así como de ciertos procedimientos de auto-recuperación.*
- *Necesario tener un diseño para aplicaciones con datos confidenciales para ofrecer seguridad. Se incorporan mecanismos de seguridad desde los niveles más bajos como el establecimiento de claves o la resistencia a la captura de un nodo, hasta los más altos del protocolo como la detección de intrusos.*
- *Para un despliegue asequible se necesitará la integración de nodos de bajo coste, auto-*

configurables y auto-organizables. En redes inalámbricas orientadas a grandes infraestructuras de redes se precisan protocolos y arquitecturas auto-organizativas.

- *Se requiere un diseño middleware eficiente, adaptable a las plataformas ya existentes, reconfigurable y escalable capaz de manejar un volumen masivo de datos sincronizados en el tiempo y organizados en bases de datos descentralizas.*
- *Para la integración con internet se dispondrá de arquitecturas escalables y protocolos eficientes. Sistemas jerárquicos y modulares pueden mejorar la flexibilidad, robustez y fiabilidad del sistema.*
- *Deberá tenerse en cuenta no sólo los costes de adquisición del producto, si no los costes de instalación, mantenimiento y operación.*
- *Se deberá tener una visualización correcta de la información en el tiempo, ya que los datos obtenidos de los clientes son muy sensibles en el tiempo. Por lo que se deberá tener unos sistemas que estén sincronizados en el tiempo. Como un preprocesamiento de los datos para poder marcar un "timestamp" en los grupos de tramas.*

Los requisitos que se precisan aquí no son fácilmente obtenibles, pero se proponen soluciones viables que solventan todos los problemas vistos previamente y, que se tratarán con profundidad a continuación en los siguientes apartados de este proyecto de fin de grado.

2 ESTADO DEL ARTE

Durante las diferentes etapas de la humanidad, ha sido siempre prioritaria la recolección del agua y su cuantificación debido a la importancia de este recurso para la mayoría de las actividades humanas.

Dada la importancia del agua, los humanos empezamos a recolectarla y a controlar su uso para la ganadería o la agricultura. Se desarrollaron diversos métodos para cuantificar el caudal usado y así tener una mejor medición del agua empleada.

Existen diversos métodos para medir el caudal de agua empleado de forma manual y varían entre sí en la calidad de la medida obtenida. Antiguamente podían obtar en métodos como soltar una hoja en el agua y dependiendo de la distancia recorrida y el tiempo que tardaba en recorrerla, obtenían una aproximación del caudal circulante, aunque no fuera muy exacta. O podían obtar por métodos más exactos como sería la instalación de una pequeña presa con una tubería y, sabiendo el diámetro de esta y con la ayuda de un cubo de capacidad conocida, calcular el caudal circulante.

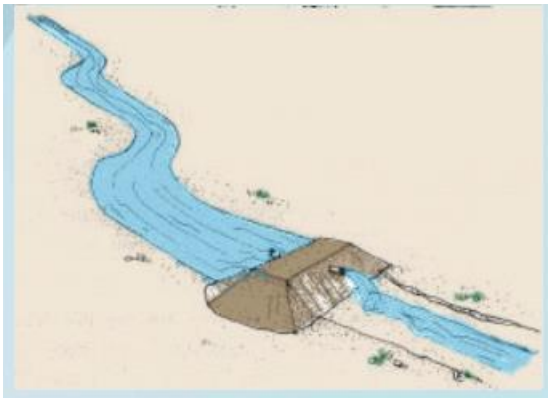


Figura 2-1. Ejemplo método de medición del caudal.

Hoy en día, los métodos convencionales de medición de agua están en desuso debido a la aparición de nuevas tecnologías que dan medidas muy fiables y exactas. Han aparecido diversidad de elementos electrónicos capaces de medir el caudal como los contadores de diferentes tipos, que reflejan la medida directamente como puede ser observado por ejemplo en los contadores de agua domésticos, que marcan el consumo de agua de un hogar.

2.1 Estado del arte en contadores.

Existen diferentes tipos de contadores de agua regulados[4] aptos para el uso agrónomo y otros diferentes para el uso doméstico. En España están regulados los diferentes tipos:

1-Chorro Único:

- Un único chorro de agua incide directamente sobre la turbina.
- La velocidad de giro de la turbina depende de la velocidad de impacto del chorro de agua(del Q circulante).

- Se instalan en posición Horizontal. Normalmente se instalan en viviendas, rara vez en regadíos.
- Trabajan para Q pequeños.
- Calibre < 20 mm. No se requieren tramos rectos de tubería aguas arriba del contador.
- Los sólidos en suspensión y sedimentaciones pueden producir sobrecontaje.

2-Chorro Múltiple

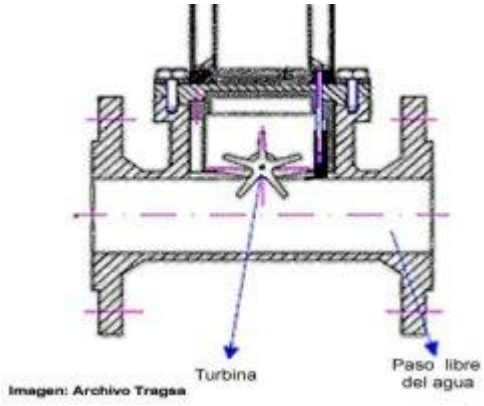
- El agua incide sobre la turbina después de haber pasado por múltiples agujeros.
- Tienen un funcionamiento más equilibrado de la turbina, por lo que se producen menos errores y tienen mayor durabilidad.
- Se instalan en posición Horizontal.
- Miden Q bajos (Diámetros < 50 mm).
- No requieren tramos rectos de tubería aguas arriba del contador.
- Las fibras, sólidos en suspensión y sedimentos provocan problemas de Sobrecontaje.
- Pérdidas de Carga: 1.5 - 2 m (para Qn).

3-Contadores Woltmann

La hélice gira en un plano perpendicular a la dirección que sigue el líquido.

- Se distinguen tres tipos según el Eje de Rotación de la Turbina.
  - Horizontal
  - Vertical
  - En codo
- Es el más habitual en regadíos debido al amplio rango de diámetros en los que se puede instalar (50-300 mm).
- Trabajan para caudales nominales entre 10 y 500 m3/h.
- Necesitan una distancia de Tramos Rectos (5 - 20) x Diámetro antes del contador.
- Es recomendable instalar filtros aguas arriba del contador ya que se evita que fibras o sólidos bloqueen la hélice.

4-Contadores Tangenciales



- Sólo parte de la Turbina recibe el impacto del agua.
- El eje de giro de la turbina es perpendicular al agua.
- No le afecta el tránsito de pequeñas partículas en suspensión ya que atraviesan el contador sin dañar la turbina.
- Tienen una precisión baja por lo que no pueden ser utilizados como elemento de facturación.
- Se deben instalar muy bien para que no den problemas: Tramos rectos aguas arriba del contador. Distancia: (15-30)x Diámetro.

5- Contadores Proporcionales

- Tiene 2 circuitos en paralelo.
- El contador se coloca en el secundario, que puede ser de chorro único o chorro múltiple.
- Tienen una precisión baja.
- No están aprobados por la Directiva 75/33/CEE.
- Deben llevar un filtro a la entrada del circuito de derivación para proteger el contador.
- No les afecta tanto la presencia de partículas en suspensión.

El problema de estos contadores es que aunque recopilen bien la información, los datos obtenidos solo pueden integrarse a base de mirar manualmente los contadores, actividad que no es económica, es lenta y requiere de muchos recursos de personal. Por lo que estos contadores al final no dan suficientes datos como para hacer gráficas de estimación de demanda o para un procesamiento de datos en general.

Esta falta de datos en las sociedades del agua causa que a la hora de una avería o de un mantenimiento, se haga costoso y difícil encontrar el origen del problema que, sumado a la antigüedad de los sistemas de distribución, causa que el tratamiento del agua para evitar perdidas sea primordial.

2.2 Estado del arte en tele-lectura.

Habiendo hablado de los sistemas de medida rutados, vamos a enfocar el progreso en el campo verdaderamente interesante para este proyecto, como es la tele-lectura de dichos contadores, las ventajas que ofrecen y la importancia que tienen a la hora de la creación de sistemas de medida inalámbricos.

Existen varios modos de tele-lectura de contadores, que nos ofrecen diferentes posibilidades a la hora del envio de datos, la distancia entre los nodos y la cantidad de dispositivos interconectados en la misma red. Estos modos se pueden clasificar por las diversas tecnologías usadas para la medida y el procesamiento de datos. Se clasifican de la siguiente forma:

- Walk By por cable directo, de la que hemos hablado anteriormente y puede conectarse por cable desde la ubicación del contador.
- Walk By por radio frecuencia de corto alcance.
- Línea telefónica RTC, GSM o GPRS. Operados mediante un moderm estándar.
- Sistemas de radio integrados en la red de las operadoras de telefonía: Narrowband IoT (**Nb-IoT**), LTE-M.
- Radio de largo alcance en banda libre: **Wireless M-Bus**, Zigbee.

Des estos mencionados para nosotros son interesantes tanto Wireless M-Bus como Nb-IoT que son los cuales se pretenden enlazar mediante un hub para aprovechar las ventajas de ambos; La conectividad entre nodos ofrecida por Wireless M-Bus y el paso de la información a internet con una sola suscrpción ofrecida por Nb-IoT.

Dado que que el tipo de tele-lectura viene dado por el tipo de comunicación inalámbrica, vamos a mostrar los tipos de comunicaciones que se disponen y las tecnologías empleadas en las redes WAN (Wireless Area Network).

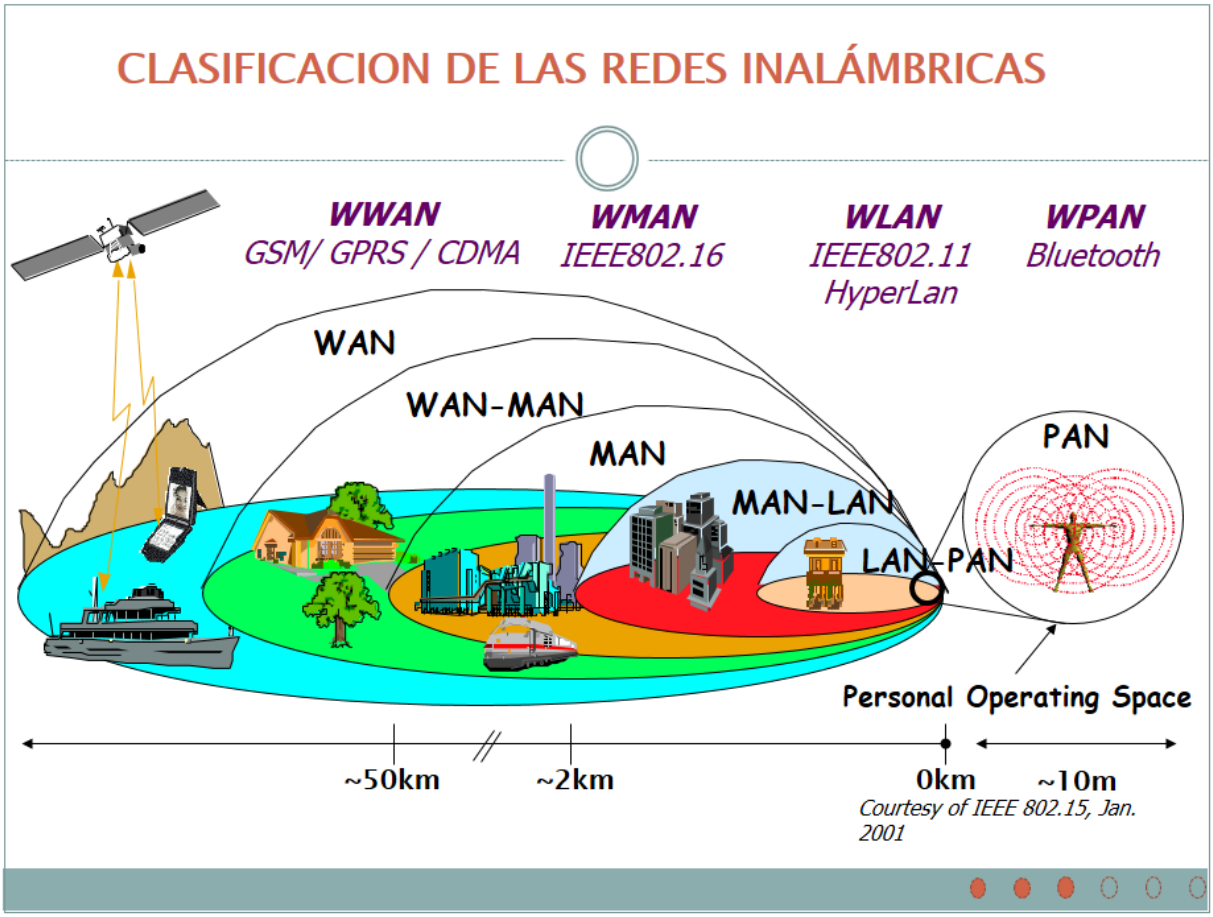


Figura 2-2. Tipos de redes usadas para IoT.

Las redes catalogadas como WPAN, son redes de muy corto alcance como Blurtooth. La distancia que recorren las señales de estas redes radica de los pocos centímetros a varios metros. Aunque útiles para otros propósitos, podemos concluir que dichas redes carecen de interés para la tele-lectura.

Las redes WLAN, las siguientes en la jerarquía de rango y basadas en IEEE802.11, como la red WiFi, tienen un mayor alcance que pueden llegar a algunas decenas de metros. Pero este tipo de redes presentan un elevado consumo y tienen muchas dificultades para superar obstáculos físicos, por lo que también limita nuestro interés para el tipo de sistema que estamos buscando. Existen ciertas variantes del 802.11 que tienen coverturas de hasta decenas de kilómetros, como las redes llamadas WiMax, pero el elevado coste de sus terminales y su consumo limitan mucho su uso para los temas relacionados con la tele-lectura.

Las redes WWAN como ZigBee-NAN o Wireless M-Bus tienen más interés para las aplicaciones de tele-lectura. Estos estándares emplean para las capas inferiores de su protocolo, en la mayor parte de los casos, el estándar IEEE 802.15.4 en algunas de sus variantes, como IEEE 802.15.4g. Con esta capa física se pueden alcanzar anchos banda suficientexº en distancias de cientos de metros con consumos muy bajos.

Es habitual la incorporación de una capa adicional superpuesta a 802.15.4, que permite conectividad IP. La capa de este estilo más empleada se denomina IETF 6LowPAN. El objetivo de esta capa es incorporar la tecnología IPv6 en redes inalámbricas IEEE 802.15.4. La ventaja principal, desde el punto de vista de la aplicación, es su capacidad de comunicar directamente con otros

dispositivos IP locales o remotos con todas las ventajas que ello conlleva: empleo de técnicas arquitecturales y de seguridad bien establecidas, modelos de datos a nivel de aplicación y servicios ampliamente extendidos, herramientas de gestión de red ampliamente difundidas y probadas, protocolos de red, etc.… Aún así, hay problemas por resolver como la sincronización de red y la imposibilidad de asegurar tiempos máximos de servicio, como en la mayor parte de estas redes.

Recientemente han aparecido en el mercado un conjunto de redes de área amplia con bajo consumo y baja tasa de datos, llamadas redes "WWANs" que están recibiendo una gran antención y se difencian en dos tipos si operan o no en banda licenciada. Están las redes cuyas redes están operadas por las telecomunicadoras, y cuyos nodos poseen una tarjeta SIM, como puede ser Nb-IoT. Y están las que operan en banda libre y pueden contar con un operador global de telecomunicaciones como SigFox, o no contar con operador como Wireless M-Bus.

A continuación, se puede observar una tabla comparativa de las diferentes redes inalámbricas.

	LoRa	EC-GSM IoT	LTE-M	NB-IoT	Wireless M-Bus
Categoría de equipo usuario LTE	N/A	N/A	Cat. M1	Cat. NB1	N/A
Rango Max. Coupling loss	<15 km 155 dB	<35 km 164 dB	<100 km 156 dB	<35 km 164 dB	<1 km 119 dB
Espectro	ISM <1GHz	Bandas licenciadas GSM	Banda licenciada LTE en-banda	Banda licenciada LTE en-banda, en banda de guarda y en banda aislada	ISM <1GHz
Ancho de banda	<500 kHz	200 kHz	1.08 MHz (1.4 MHz de ancho de banda de la portadora)	180 kHz (200 kHz de ancho de banda de portadora)	868 MHz (Depende del modo de funcionamiento)
Máxima tasa de datos	< 50 kbps (DL/UL)	< 140 kbps (DL/UL)	< 1 Mbps (DL/UL)	<170 kbps (DL) <250 kbps (UL)	< 100 kbps (DL/UL)
Disponibilidad	SI	En breve	Desconocido por la fata de silicio	Estándar aún no terminado	SI

Finalmente, decir que las tecnologías 5G van a traer en muy pocos años, nuevas posibilidades de acceso de gran calidad, elevado ancho de banda y muy bajo consumo que pueden cambiar de manera drástica la forma en que hoy conocemos la tele-lectura.

Por último, mencionar la gran ventaja que posee Wireless M-Bus sobre por ejemplo SigFox para la aplicación que nosotros queremos construir. Por ejemplo, para la lectura de un cuarto de contadores de un edificio, se necesitaría una conexión por cada uno de los meters para que enviasen datos de forma inalámbrica. Esto en SigFox significaría una suscribción por cada uno de los dispositivos que quieren mandar datos.

Sin embargo, la misma tarea realizada con Wireless M-Bus, ahorraría muchos costes, a la hora de que todos esos meters podrían mandar la información por este protocolo a un concentrador, que sería el responsable de enviar la información y sí tendría suscripción. Pero mientras que, de la primera forma, su hubiera 30 contadores necesitaríamos 30 suscribciones, y utilizando Wireless M-Bus aunque haya 30 contadores solo necesitaríamos una suscribción. Esto facilita y mejora enormemente los procedimientos de medida en las ciudades con gran densidad de bloques de pisos.

3 WIRELESS M-BUS

El protocolo M-Bus (Meter Bus) corresponde a un protocolo de comunicación estándar en Europa[5] para la lectura de dispositivos de medición. Este sistema de comunicación se basa en el sistema jerárquico maestro/esclavo (Master/Slave), actuando siempre como Master, y está normalizado según la norma EN13757. Se trata de un sistema de bus mono-master y half-duplex, lo que significa que existe un solo Master conectado a distintos esclavos usando comunicación bidireccional. Es un protocolo utilizado para la lectura de medidores de calefacción, **contadores de volumen de agua** y eléctricos, entre otros. En este protocolo, se nombra a los esclavos como "Meters", responsables de recolectar la información de los sensores y mandarla hacia los nodos maestros, llamados "Collector" pudiendo ser concentradores multi-tarea, recibidores estacionarios, o componentes de un sistema de red. Esta configuración favorece las topologías de red asimétricas con medidores de bajo consumo por una parte y unos receptores de más alto nivel en el otro lado.

Además este sistema de comunicación cuenta con la variante inalámbrica Wireless M-Bus, de manera que en una misma instalación se pueden optar utilizar el cableado o la radiofrecuencia.

Como se ha dicho ya, Wireless M-bus es la alternativa inalámbrica del protocolo de comunicación M-Bus. Este tipo de sistema cuenta con distintos modos (S1/S2, T1/T2, C1/C2, N1/N2) para el envio y recepción de los datos medidos. Esta instalación nos permite el uso del protocolo M-Bus con la ventaja de poder usar redes de mayor alcance gracias a la radiofrecuencia.

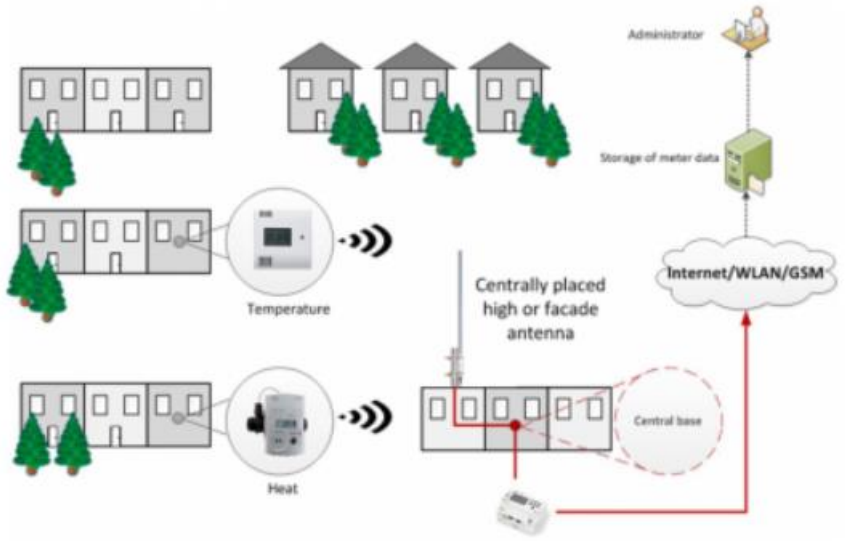


Figura 3-1. Ejemplo red usando Wireless M-Bus.

Los diferentes modos mencionados encima corresponden a las diferentes posibilidades de funcionamiento del protocolo Wireless M-Bus. Los diferentes modos de funcionamiento varían en la

forma y frecuencia de envio de los datos.

-Modo S1, S2: Es el "**S**tationary mode". Los nodos "meter" envían datos muchas veces al día, y el nodo "collector" ahorra energía manteniéndose dormido hasta recibir una señal de "wake up" de los nodos "meter" antes del envío de datos.

-Modo T1, T2: Se corresponde con el "Frequent **T**ransmit mode". Los nodos "meter" envían datos de forma periódica a los nodos "collector" dentro de su rango. El intervalo de tiempo entre cada envío de datos puede configurarse en términos de varios segundos o minutos.

-Modo C1, C2: Se corresponde con el "**C**ompact mode". Es similar al modo de transmisión T pero en este caso, permite el envío de más datos dentro del mismo consumo de energía y el mismo Duty Cycle.

-Modo N1(a-f), N2(a-f): Se corresponde "**N**arrowband mode". Se usa para transmisiones de alto rango en bajo consumo.

Dentro de los diferentes modos de comunicación, se puede diferenciar también entre comunicación unidireccional y bidireccional.

-Unidireccional: Sólo permite el envio de datos de los nodos "meter" a el/los nodos "collector". La ventaja del uso de este modo de funcionamiento consiste en su básica implementación. Ya que los nodos esclavos sólo tienen que enviar los datos y los nodos maestros recibirlos.

-Bidireccional: En este modo también pueden enviarse datos desde los nodos "collector" hacia los nodos "meter". Para el uso de este modo, ambos tipos de nodos tienen que estar configurados de forma bidireccional, un nodo "collector" no puede recibir información de un nodo "meter" configurado de forma unidireccional.

El protocolo Wireless M-Bus puede operar con diferentes tipos de tramas, tipos de codificación de datos, frequencias de modulación (como Frequency Shift Keying (FSK) o Gaussian Frequency Shift Keying). En la siguiente tabla se pueden observar las diferentes configuraciones.

Mode	Meter	Collector	Data Rate	Encoding	Modulation	Frequency [MHz]
N1a, N2a	RX, TX	RX, TX	4.8 kcps	NRZ	GFSK	169.406250
N1b, N2b	RX, TX	RX, TX	4.8 kcps	NRZ	GFSK	169.418750
N1c, N2c	RX, TX	RX, TX	2.4 kcps	NRZ	GFSK	169.431250
N1d, N2d	RX, TX	RX, TX	2.4 kcps	NRZ	GFSK	169.443750
N1e, N2e	RX, TX	RX, TX	4.8 kcps	NRZ	GFSK	169.456250
N1f, N2f	RX, TX	RX, TX	4.8 kcps	NRZ	GFSK	169.468750
T2	RX	TX	32.768 kcps	Manchester	FSK	868.30
T1, T2	TX	RX	100 kcps	3-Out-Of-6	FSK	868.95
S1, S2	RX, TX	RX, TX	32.768 kcps	Manchester	FSK	868.30
C2	RX	TX	50 kcps	NRZ	GFSK	869.525
C1, C2	TX	RX	100 kcps	NRZ	FSK	868.95

Antes de pasar a los diferentes tipos de contadores que interesan en este proyecto y los tipos de

tramas que se pueden enviar, se va a dar a ver una pequeña introducción del Stack que se usa.

En este caso, se ha utilizado el Software Stack de **STACKFORCE**. Este consiste en una arquitectura de 6 capas como se puede ver en la siguiente imagen.

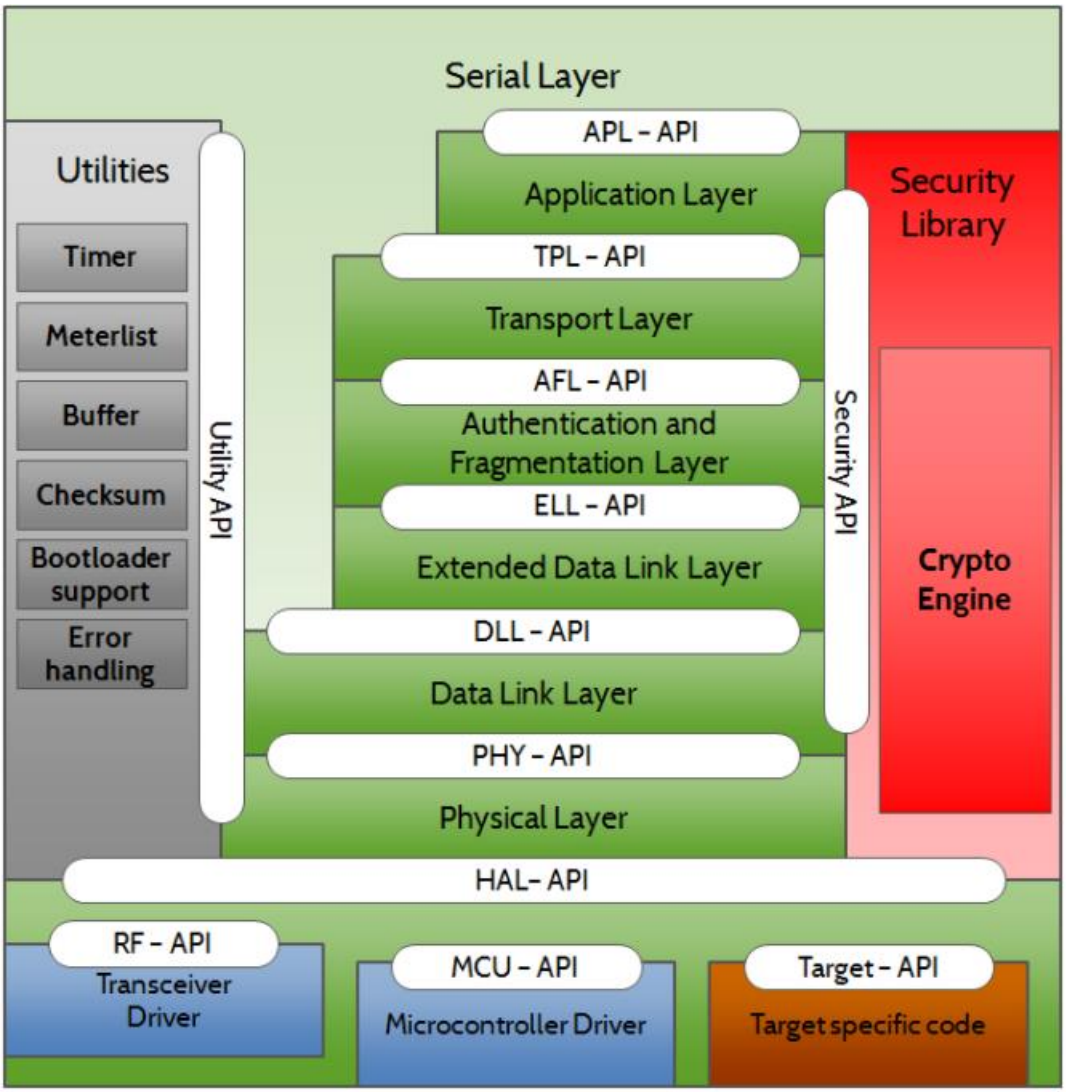


Figura 3-2. Arquitectura de Wireless M-Bus del SW Stack de STACKFORCE.

Estas capas son la capa fisisca (**Physical Layer**), la capa de envio de datos (**Data Link Layer**), la capa de envio de datos extendida (**Extended Data Link Layer**), la capa de autentificación y fragmentación (**Authentication and Fragmentation Layer**), La capa de transporte (**Transport Layer**) y la capa de aplicación (**Application Layer**). Además de mencionar la HAL (**Hardware Abstraction Layer**) encargada de extraer los recursos de HW necesarios para el correcto funcionamiento del Stack.

Además el Stack contiene los drivers necesarios para la inicialización de los mñódulos HW dentro de los dispositivos de TI que soportan el Wireless M-Bus Stack. Consta también con una interfaz llamada “Crypto-engine” que se basa en un conjunto de librerías necesarias para controlar la seguridad del protocolo M Bus.

3.1 Tipos de contadores

Existen diferentes tipos de contadores de agua en el mercado. Estos están caracterizados por varios factores, entre ellos la forma que tiene cada uno de ellos en realizar la medida. Nosotros vamos a diferenciarlos entre electrónicos y mecánicos. Cada uno de ellos también se diferencian en los tipos de tramas que pueden manejar.

Dato obtenido (Trama)	Electrónico	Mecánico
Número serie del contador (A, A+, B)	✓	✗
Valor de la medida (A,A+,B)	✗	✓
Unidad de la medida (A, A+)	✓	✗
Factor de potencia (A+)	✓	✗
Unidad de tiempo (A+)	✓	✗
Posible fuga o fraude en instalación (A, A+)	✓	✗
Código de diagnósticos y error (A+)	✓	✗
Estado batería del contador (A ,A+)	✓	✗

Figura 3-1-1. Información obtenida en función del contador y la trama.

3.2 Tipos de tramas

Para el protocolo de Wireless M-Bus existen 4 tipos diferentes de tramas[6].

Nombre	*Sentido de la comunicación*	*Velocidad*
A	Del dispositivo hacia el SADC	1 200 baud
A+	Del dispositivo hacia el SADC	1 200 baud
B	Bidireccional	1 200 baud
C	Bidireccional	Programable. 1 200 baud al inicio de sesión

Figura 3-2-1. Tipos de tramas.

Los dispositivos implementarán o bien la trama A o la trama A+ para el inicio de sesión. La implementación de las tramas bidireccionales B y C es opcional para el fabricante del dispositivo. Solo mencionar que para el uso de la trama bidireccional C el dispositivo deberá tener implementada previamente la trama A+. Vemos en la siguiente imagen un diagrama de funcionamiento de las tramas.

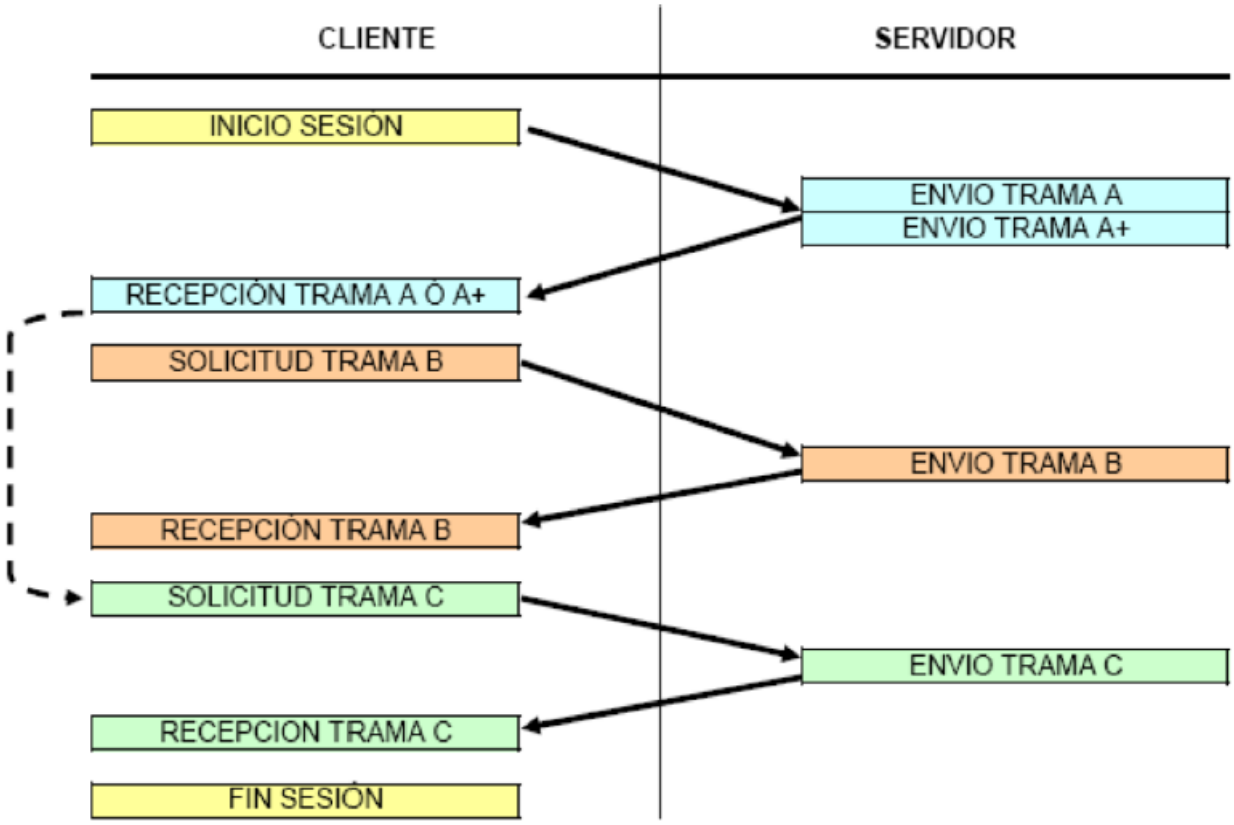


Figura 3-2-2. Ejemplo de funcionamiento de las diferentes tramas.

En la siguiente tabla se presenta de forma resumida las características de las diferentes tramas y en los siguientes puntos se detallará cada una de ellas.

A	A+	B	C
Del dispositivo hacia el SADC	Del dispositivo hacia el SADC	Bidireccional	Bidireccional
Transmisión asíncrona modo byte	Transmisión asíncrona modo byte	Transmisión asíncrona modo byte	Transmisión asíncrona modo byte
1 bit de Start	1 bit de Start	1 bit de Start	1 bit de Start
1 200 baud	1 200 baud	1 200 baud	V programable, 1 200 baud al inicio de la sesión
7 bits de datos código ASCII	7 bits de datos código ASCII	7 bits de datos código ASCII	8 bits de datos código ASCII
No paridad	No paridad	No paridad	No paridad
2 bits de stop	2 bits de stop	2 bits de stop	1 bit de stop
32 bytes	Máx 128 bytes	Máx 128 bytes	La longitud total es NID+NPayLd+8. La longitud mínima es 8 y la longitud máxima codificable es 8+127(NID)+1023(NPayLd)=1158

Figura 3-2-3. Resumen funcionamiento de las tramas.

3.2.1 Trama A

La trama A envía información en un formato de 32 bytes asíncronos de la siguiente manera.

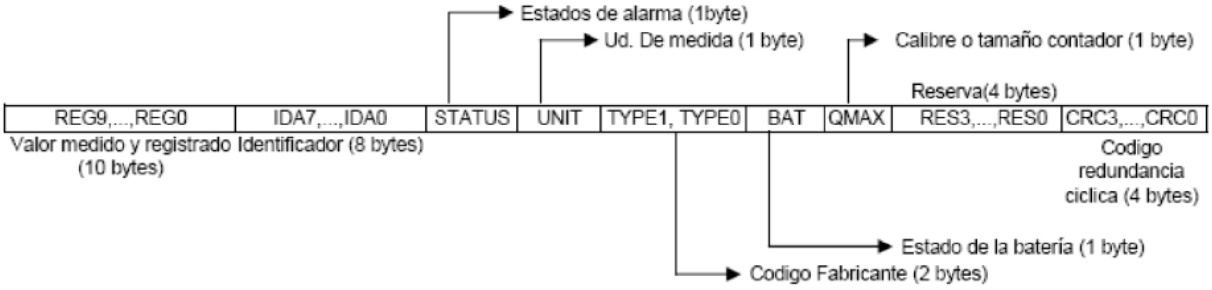


Figura 3-2-1-1. Formato envio de datos tramas A.

La información de cada uno de los campos de la trama A se registra en esta tabla.

Campos de la TRAMA A		
Campo	**Longitud en bytes**	**Descripción**
REG	10 MSB primero	Valor medido y registrado por el dispositivo, contiene siempre 10 dígitos decimales, de REG9 a REG0, enviando en primer lugar REG9. El valor de cada uno de los dígitos depende de la unidad de medida.
IDA	8 MSB primero	Identificador absoluto o número de serie de 8 dígitos hexadecimales, IDA7 A IDA0. Se envía en primer lugar IDA7.
STATUS	1	Un byte de estados de alarma del dispositivo, utilizando los cuatro bits menos significativos.
UNIT	1	Codificación de la unidad de medida y transmisión. Se utiliza conjuntamente con REG para conocer la cantidad medida.
TYPE	2 MSB primero	Codificación del fabricante, dos bytes, TYPE1 (primer byte enviado) y TYPE0.
BAT	1	Estado de la pila, en un dígito decimal.
SIZE	1	Codificación del calibre o tamaño del dispositivo.
RES	4	Reservados para aplicación propia del dispositivo, 4 bytes de RES3 (primero en ser enviado) a RES0.
CRC16	4 MSB primero	Código de redundancia cíclica en cuatro bytes, de CRC3 a CRC0, enviando en primer lugar CRC3. El CRC-16 se calcula desde el primer byte REG9 a RES0.

Cada byte de las tramas se envía de forma asíncrona con 1 bit de “start”, 7 bits de datos codificados en ASCII, sin bit de paridad, 2 bit de “stop” y a 1200 baudios.

El bit menos significativo (LSb) se transmite primero, como en las UART y en la mayoría de sistemas de transmisión serie.

Algunos de los bytes contienen información de un dígito decimal (REG, BAT), otros pueden ser hexadecimales o decimales (IDA, UNIT, TYPE, SIZE, CRC) y el STATUS contiene la información de 4 bits independientes. En el caso de trasmitir un dígito decimal el dispositivo añadirá a su código BCD los bits 011b en su parte más significativa, convirtiéndolo en ASCII. En el caso de transmitir una cifra hexadecimal o el STATUS también se añaden 011b, transformando la información en un

formato ASCII desde 30h a 3Fh.

Ahora se expondrá una breve descripción de la información que se envía en la trama A:

-REG: Valor medio registrado. *Contiene siempre 10 dígitos decimales.*

-IDA: Número de serie. *Número de fabricación, contiene siempre 8 dígitos hexadecimales.*

-STATUS: Estado del dispositivo. *Contiene 011b en los 3 bits más significativos y 4 bits individuales.*

-UNIT: Unidad de medida y transmisión. *Se utiliza conjuntamente con REG para saber la cantidad y la unidad de la medida.*

-TYPE: Información del fabricante. *Dos bits que codifican el fabricante del dispositivo.*

-BAT: Estado de la pila.

-SIZE: Codificacion del calibre/tamaño.

3.2.2 Trama A+

La mejora de las tramas A+ respecto de las A es que disponen de una codificación menos rígida y sin limitación de tamaño en ciertos casos. Mezclar dispositivos que se envíen entre sí tramas A y A+ no es un problema ya ue decodificando el primer carácter ya sabemos a cual de ellas pertenece. Las tramas A+ se envían a la misma velocidad que las tramas A, 1200 baudios, disponen de 7 bits de datos, sin paridad y 2 bits de "stop".

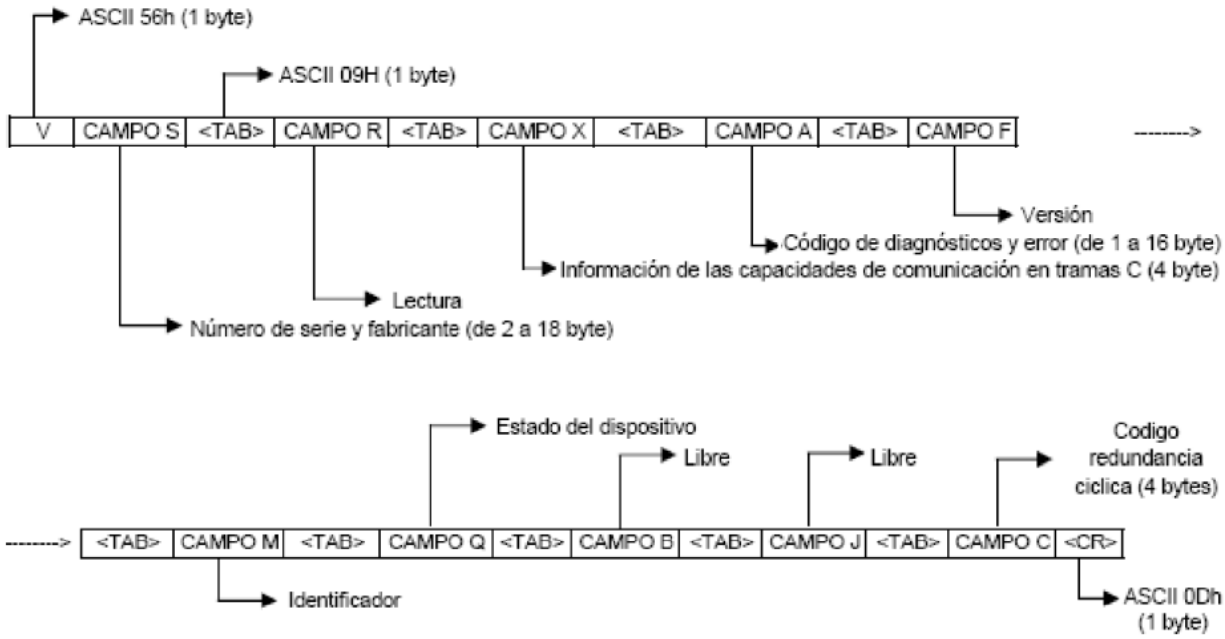


Figura 3-2-2-1. Formato envio de datos tramas A+

Donde la siguiente tabla caracteriza cada uno de los campos vistos en la imagen anterior.

Campos de la TRAMA A+			
Campo	**Longitud**	**Obligatoriedad**	**Descripción**
V	1		Carácter "V" mayúscula (ASCII 56h)
Campo S	De 2 a 18 MSB primero	Obligatorio	Fabricante y número de serie. Debe ser el primer campo.
TAB	1	Obligatorio	Carácter tabulador (ASCII 09h), utilizado para separar campos.
Campo R	Sin especificar	Obligatorio	Lectura. Incluye valor, unidad, factor de potencia y unidad de tiempo.
Campo X	4	Obligatorio	Información de las capacidades de comunicación en tramas C.
Campo A	De 1 a 16	Opcional	Código de diagnósticos y error, específico del fabricante, de 1 a 16 caracteres ASCII. Cada uno de ellos se forma a partir de un byte 0xxxxxxb, al que se le suma 30h en el momento de la transmisión, donde cada x representa un bit.
Campo F	Sin especificar	Opcional	Versión, específico de cada fabricante.
Campo N	Sin especificar	Opcional	Identificador del modelo, en texto libre.
Campo Q	Sin especificar	Opcional	Estado del dispositivo.
Campo B	Sin especificar	Opcional	Identificador de cliente, usuario o Nº Serie Ampliado. Opcional.
Campo J	Sin especificar	Opcional	Texto libre.
Campo C	4 MSB primero	Obligatorio	CRC16 de la trama, sin incluir el propio campo C, ni el primer carácter "C" de este campo. El primer carácter de la trama "V" queda incluido en el cálculo. Este campo debe ser el último.

Hay diferencias a la hora de leer las tramas A+ en un solo dispositivo o si tenemos varios de ellos. Se priorizará la lectura en varios, ya que va a ser lo que se está investigando en este proyecto.

El protocolo para leer dos o más dispositivos por el bus será explicada por fases inmediatamente después, pero es remarcable decir que el orden de los dispositivos enviando información es aleatorio.

- El SADC activa la señal SEL.
- Los dispositivos detectan dicha activación y esperan un tiempo, tras el cual, si la señal SEL sigue activada, envían la trama de formato A+ (o A). Ese tiempo de espera, que está comprendido entre 0s y 0.75s, es un retardo que implementa el dispositivo con la finalidad de evitar colisiones, calculándose mediante un algoritmo de generación de números pseudo-aleatorios, con las siguientes características:
- El retardo pseudo-aleatorio deberá tener una distribución uniforme en un intervalo de tiempo entre 0s y 0,75s.

- La probabilidad de que los dispositivos distinitos apliquen retardos pseudo-aleatorios distanciados menos de 25 micro-segundos deberá ser menor al 0.1%.
- El dispositivo que calcula un retardo menor es el primero que consulta la señal SEL tras dicho retardo, encontrándola todavía activada y comenzando a transmitir.
- El SADC al detectar el bit de arranque de una trama transmitida por uno de los dispositivos en la línea SDATA, desactiva la señal SEL, en un tiempo no superior a 25 micro-segundos. Este el método de arbitraje del bus, de forma que se evita que otros dispositivos transmitan a la vez y se produzcan colisiones.
- Los demás dispositivos, que han calculado un retardo mayor al que ha comenzado a transmitir, encuentran la señal SEL desactivada y vuelven al estado inicial.
- Una vez terminada la recepción de un dispositivo, el SADC vuelve a activar la señal SEL. Los dispositivos que habían quedado en espera de su turno, cuando vuelven a detectar la señal SEL activada, comienzan un nuevo proceso de cálculo del retardo aleatorio previo a intentar transmitir su trama A o A+. Los dispositivos que ya han transmitido quedan a la escucha.
- En todo caso, una vez que el dispositivo detecta la desactivación de la señal SEL pone en marcha un timeout de 1.25s para cerrar la sesión de comunicación. Por lo tanto el SADC debe asegurarse de que la señal no permanezca más de 1.25s seguidos (incluyendo el tiempo de transmisión del dispositivo), de lo contrario algunos dispositivos podrían dar por terminada la sesión de comunicación por timeout, enviando la trama de nuevo al detectar posteriormente la señal SEL activada.
- En la práctica, si no se conoce a priori el numero de dispositivos presentes en el bus, el SADC debe mantener la señal SEL activa durante un tiempo superior a 0.75s para estar seguro de que no quedan dispositivos por enviar su trama A o A+. Se recomienda a los desarrolladores del SADC que utilicen un tiempo de 1.5s para cubrir cualquier contingencia que pueda ocurrir durante la lectura en la instalación real.
- Para cerrar la sesión de comunicación la señal SEL deberá desactivarse durante al menos 2s, de esta manera asegura que en todos los dispositivos incluyendo los que estaban calculando el retardo aleatorio en el momento de desactivarse la señal SEL, el timeout les ha hecho salir del modo de comunicación.

3.2.3 Trama B

Las tramas de formato A o A+ transmiten información básica del dispositivo. Son tramas de un solo sentido, del dispositivo hacia el SADC. Las tramas B son bidireccionales, transmiten información del SADC al dispositivo y viceversa. Estas tramas se emplean para acceder al resto de funciones del dispositivo como conocer la versión del firmware del dispositivo. No se profundizará mucho en estas tramas debido a que no requieren tanta importancia para este proyecto.

El formato de mensaje de estas estructuras es el siguiente:

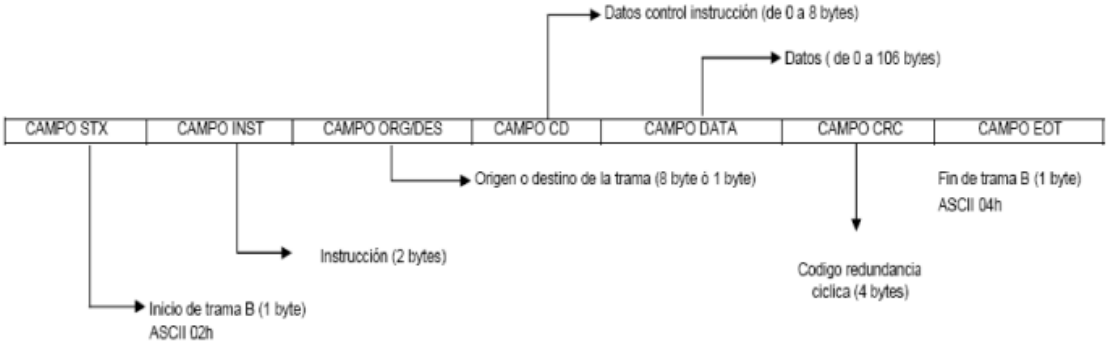


Figura 3-2-3-1. Formato envio datos tramas B.

Cada trama B tiene una estructura muy similar a las tramas A. Los datos se transmiten asíncronamente modo byte a la misma velocidad de 1200 baudios, 7 bits de datos, sin paridad y 2 bits de "stop". Y su estructura está recogida en la siguiente tabla.

Campo	Longitud en bytes	Codificación	Descripción
STX	1	000 0010b	Inicio de trama, carácter STX, ASCII 02h. Imposible en una trama A o A+.
INST	2	011b-INST1 011b-INST0	Dos caracteres hexadecimales, INST1 (primero en ser enviado) e INST0, que indican la instrucción. Cada carácter es enviado en un byte, con la parte alta del mismo igual a 011b.
ORG/DES	1 / 8 MSB primero	011b-IDa7 … 011b-IDa0	Indica el origen o destino de trama: cuando va del dispositivo al SADC indica origen, cuando va del SADC al dispositivo indica el destino. Está formado por 8 caracteres: IDA7 (primero en ser enviado) a ID0. Cada byte con la parte alta igual a 011b.
CD	De 0 a 8	011b-CDm … 011b-CD0	Datos de control, longitud dependiente de la instrucción, desde CD0 a CDn, en el mismo formato que los campos anteriores. Este campo puede no existir.
DATA	De 0 a 106	011b-Dn ... 011b-D0	Datos, longitud dependiente de la instrucción, desde D0 a Dn, en el mismo formato que los campos anteriores.
CRC	4	011b-CRC3 … 011b-CRC0	CRC-16, según algoritmo indicado para la trama A, incluye para el cálculo los campos desde <INST> a <DATA>, ambos inclusive.
EOT	1	000 0100b	Final de trama, carácter EOT, ASCII 04h

Pero las tramas B son capaces de recibir información del dispositivo también y pueden mandar peticiones y mensajes. Estas tramas en formato B enviadas por el dispositivo siguen el mismo protocolo de escucha del SEL para evitar colisión utilizados por las tramas en formato A. Pueden enviarse los siguientes mensajes.

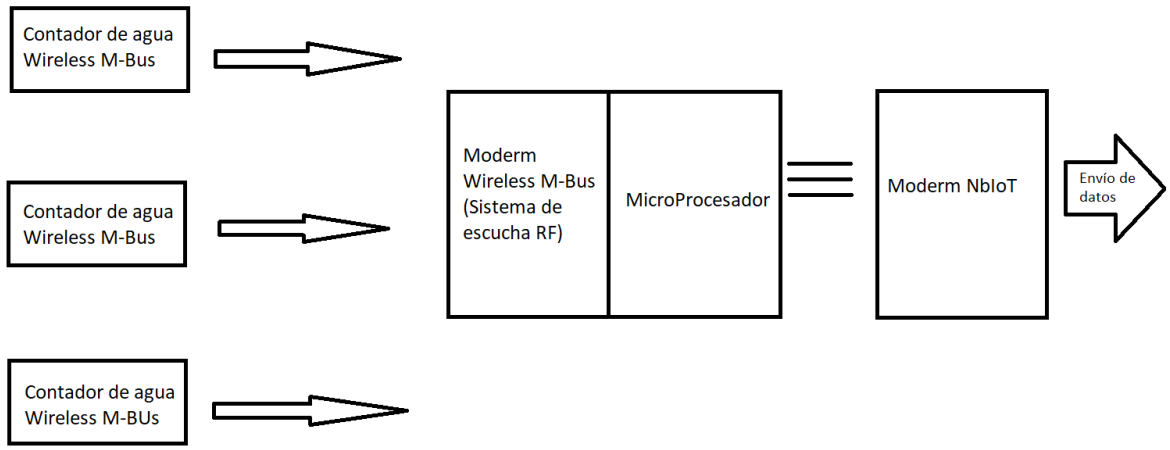
Instrucciones en la TRAMA B					
Nombre instrucción	**Código**	**Longitud campo CD**	**Longitud campo datos hacia dispositivo**	**Longitud campo datos hacia SADC**	**Descripción**
RVER	14	0	0	4	Leer el número de código (dos caracteres) y la versión del firmware (dos caracteres: x.x)
REXT	12	0	0	48	Enviar información extendida. El contenido de esta información extendida dependerá de cada dispositivo
WEXT	13	1	16	0	Escribir el área de información extendida. Esta instrucción no tiene contestación del dispositivo
INTACT	1C	5	Variable	Variable	Acceso lectura/escritura al mapa de memoria
GOW2	02	0	0	0	Pasar a estado W2 (véase el capítulo relativo a la capa de sesión). Esta instrucción no tiene contestación del dispositivo hacia el SADC
GOW0	03	0	0	0	Pasar a estado W0, si no estaba en W2 (véase el capítulo relativo a la capa de sesión). Esta instrucción no tiene contestación del dispositivo hacia el SADC

4 ARQUITECTURA

En este apartado se comentará la arquitectura del sistema que se quiere diseñar, esta tiene varias fases. Primero los contadores con Wireless M-Bus que envían de forma inalámbrica a través de este protocolo hacia el concentrador.

El concentrador que ha recibido la información de los contadores a través de la radio frecuencia, comunica los datos y los commandos recibidos al micro procesador. El micro procesador, que tiene cargado previamente un código principal y una serie de drivers, ayudan a la comprensión de las órdenes y a la gestion de dicha información.

Una vez procesada esta información, vuelve a ser traducida a comandos AT, pero en esta ocasión para su interpretación por parte del moderm Nb-IoT que se encargará de procesar los datos recibidos por UART desde el micro procesador y subirlo a la red.



Ahora vamos a profundizar qué es lo que ocurre dentro del microprocesador y cómo se evaluan e interpretan los datos.

Para esta evaluación de la información, serán necesarios diferentes archivos. Se necesitará un programa principal o “main.c” que será el archio que recoge todas las librerias y comandos principals a realizar por el microprocesador.

Este main.c será escrito en base al “Wireless M-Bus Software Stack” que está distribuido por Texas Instruments®. En este stack se incluyen todas las librerias necesarias para el correcto conexionado de los dispositivos, la transmission de información entre ellos y la configuración de todos los parámetros necesarios.

Este stack, además opera con “RTOS”, una herramienta muy útil para el manejo de la información en micro controladores en tiempo real, sus siglas significan “Real Time Operating System”. Esta configuración también está definida con sus respectivas librerias en el stack proporcionado por Texas instrument.

Por lo que nos quedan los drivers para interpretar la información recibida por el moderm Wireless M-Bus, y el que envía los comandos necesarios para el envio de información al moderm Nb-IoT.

Main.c
Driver Wireless M-Bus
Driver Nb-IoT
Wireless M-Bus Software Stack
RTOS

Donde los dos campos coloreados de azul son el objeto a desarrollar en este proyecto, ya que el Stack y el RTOS los prove Texas Instruments® y el Main.c es un Desarrollo más complejo y escapa a los objetivos de este proyecto de fin de grado.

5 PLATAFORMA HARDWARE

En este proyecto se pretende utilizar un elemento HW dedicado a la conexión Wireless de los contadores de agua y la respectiva subida de datos a un servidor para su posterior procesado y manejo de información.

5.1 Requisitos Hardware.

Para el uso del protocolo Wireless M-Bus, necesitaremos ciertas características en el HW que soporten el protocolo utilizado. Estas se pueden visualizar en la siguiente tabla.

Microcontrolador para el procesado de la información.	¿?
Modo de bajo consumo.	¿?
Modulo de seguridad.	¿?
UART, I2C y RTC.	¿?
Ancho de banda soportable para el protocolo	¿?
Fabricante de la plataforma HW proporciona la pila de protocolos Wireless M-Bus	¿?
Moderm compatible con las bandas de frecuencia Wireless M-Bus	¿?
Moderm compatible con las modulaciones Wireless M-Bus	¿?

Ahora se investigará para comprobar qué módulo cumople con los requisitos anteriormente marcados, y si los satisface, elegirlo para su uso.

5.2 Plataforma HW elegida.

Después de la investigación de varios elementos se ha concluido en utilizar el microcontrolador **CC1310 SimpleLink™ Ultra-Low-Power Sub-1 GHz Wireless MCU.** Este dispositivo pertenece a la familia de Texas Instruments CC26xx y CC13xx[7] de dispositivos de RF de 2.4 GHz de consumo de energía ultra bajo. Una corriente del MCU y RF activa muy baja y un consumo de corriente con modo de potencia baja que ofrecen una excelente duración de la batería. Además de poder operar con baterías de celda tipo moneda y en aplicaciones de energy-harvesting.

El dispositivo es el primer integrante de una familia de Sub-1 GHz de MCU inalámbricos, rentables y de consumo ultra bajo. El dispositivo CC1310 combina un transceptor RF de consumo muy bajo y flexible con un poderoso microcontrolador Cortex-M3 de 48 MHz en una plataforma que admite varias capas físicas y estándares de RF. Un controlador de radio Cortex-M0 dedicado maneja

los comandos de protocolo RF de bajo nivel que se almacenan en ROM o RAM, por tanto, asegura esa flexibilidad y bajo consumo.

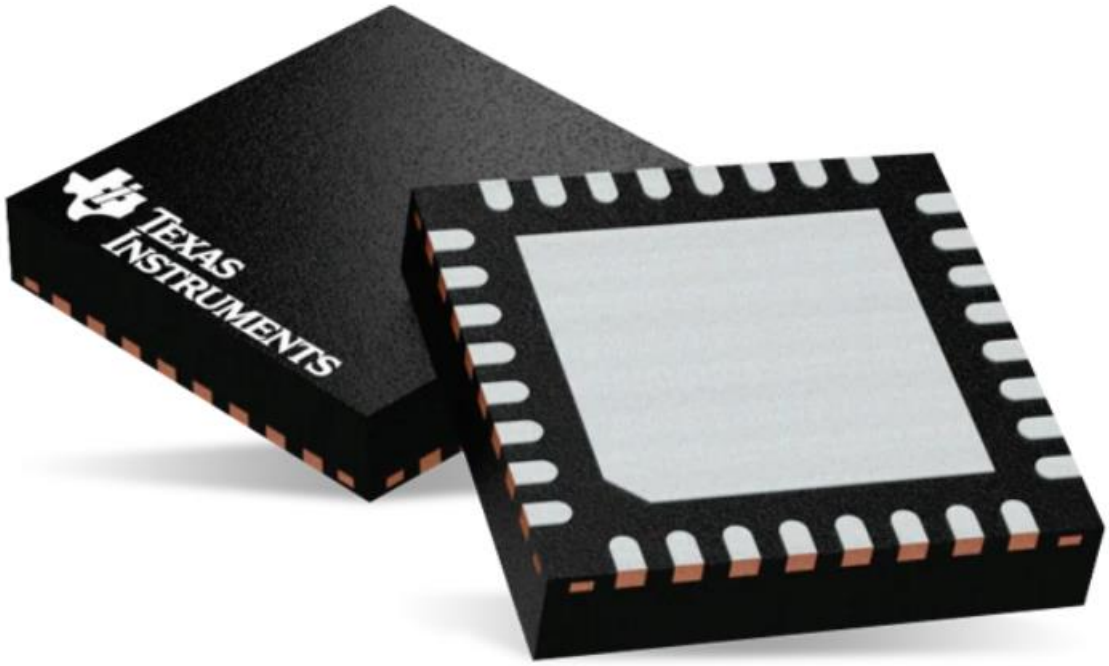


Figura 5-1. CC1310 SimpleLink™ Ultra-Low-Power Sub-1 GHz Wireless MCU.

Las características del dispositivo son las siguientes:

- Microcontrolador
  – Procesador potente Arm® Cortex® -M3.
  – EEMBC CoreMark® Score: 142.
  – EEMBC ULPBench™ Score: 158.
  – Velocidad del reloj hasta 48-MHz.
  – 32KB, 64KB, and 128KB de flash programable in-system.
  – 8KB of SRAM para Cache (o RAM de propósito general)
  – 20KB de SRAM para Fuga-Ultra-Baja.
  – 2-Pin cJTAG y JTAG Debugging.
  – Soporte actualizado Over-the-Air (OTA).
- Controlador del sensor de Ultra-Bajo-Consumo
  – Puede funcionar de forma autónoma al resto del sistema.
  – Arquitectura de 16-Bit.
  – 2 KB de SRAM de Fuga-Ultra-Baja para codificación y datos.

- Arquitectura eficiente, colocando partes del TI-RTOS, Drivers y Bootloader en la ROM
- Paquete RoHS-Compliant.
  – 7-mm × 7-mm RGZ VQFN48 (30 GPIOs).
  – 5-mm × 5-mm RHB VQFN32 (15 GPIOs).
  – 4-mm × 4-mm RSM VQFN32 (10 GPIOs).
- Periféricos.
  – Todos los pines digitales periféricos pueden rutarse a cualquier GPIO.
  – Cuatro módulos de tiempo de uso general (ocho 16-Bit o cuatro 32-Bit Timers, PWM cada uno).
  – 12-Bit ADC, 200 ksamples/s, 8-Channel Analog MUX.
  – Comparador en tiempo continuo.
  – Comparador con reloj de Ultra-Bajo-Consumo.
  – Fuente de corriente programable.
  – UART.
  – 2× SSI (SPI, MICROWIRE, TI).
  – I 2C, I2S.
  – Real-Time Clock (RTC).
  – AES-128 Security Module.
  – Generador real de números aleatorios (TRNG)
  – Soporte para ocho sensores capacitivos.
  – Sensor de temperatura integrado.
- Sistema Externo
  – Convertidor DC-DC en el chip.
  – Integración “sin costura” con el SimpleLink™ CC1190 Range Extender.
- Bajo consumo
  – Rango de voltaje de entrada: 1.8 to 3.8 V.
  – RX: 5.4 mA.
  – TX at +10 dBm: 13.4 mA.
  – Active-Mode MCU 48 MHz Running Coremark: 2.5 mA (51 µA/MHz).
  – Active-Mode MCU: 48.5 CoreMark/mA.
  – Active-Mode Sensor Controller at 24 MHz: 0.4 mA + 8.2 µA/MHz.
  – Controlador del sensor, Cada segundo despierto hace un 12-Bit ADC Sampling: 0.95 µA.
  – Standby: 0.7 µA (RTC Running and RAM and CPU Retention).
  – Shutdown: 185 nA (Despertar con eventos externos).
- Sección RF
  – Sensibilidad del recibidor excelente –124 dBm usando un modo de rango largo, –110 dBm

at 50 kbps.

– Sensibilidad excelente (±100 kHz): 56 dB.

– Excellent Blocking Performance (±10 MHz): 90 dB.

– Salida de potencia programable hasta +15 dBm.

– Single-Ended or Differential RF Interface.

– Suitable for Systems Targeting Compliance With Worldwide Radio Frequency Regulations – ETSI EN 300 220, EN 303 204 (Europe) – FCC CFR47 Part 15 (US) – ARIB STD-T108 (Japan).

– Wireless M-Bus (EN 13757-4) and IEEE® 802.15.4g PHY.

- Entorno de desarrollo y herramientas.

  – Full-Feature and Low-Cost Development Kits.

  – Diseños de referencia múltiples para configuraciones RF diferentes.

  – Packet Sniffer PC Software.

  – Sensor Controller Studio.

  – SmartRF™ Studio.

  – SmartRF Flash Programmer 2.

  – IAR Embedded Workbench® for Arm.

  – Code Composer Studio™ (CCS) IDE.

  – CCS UniFlash.

El diagrama de flujo del dispositivo se corresponde con el siguiente.

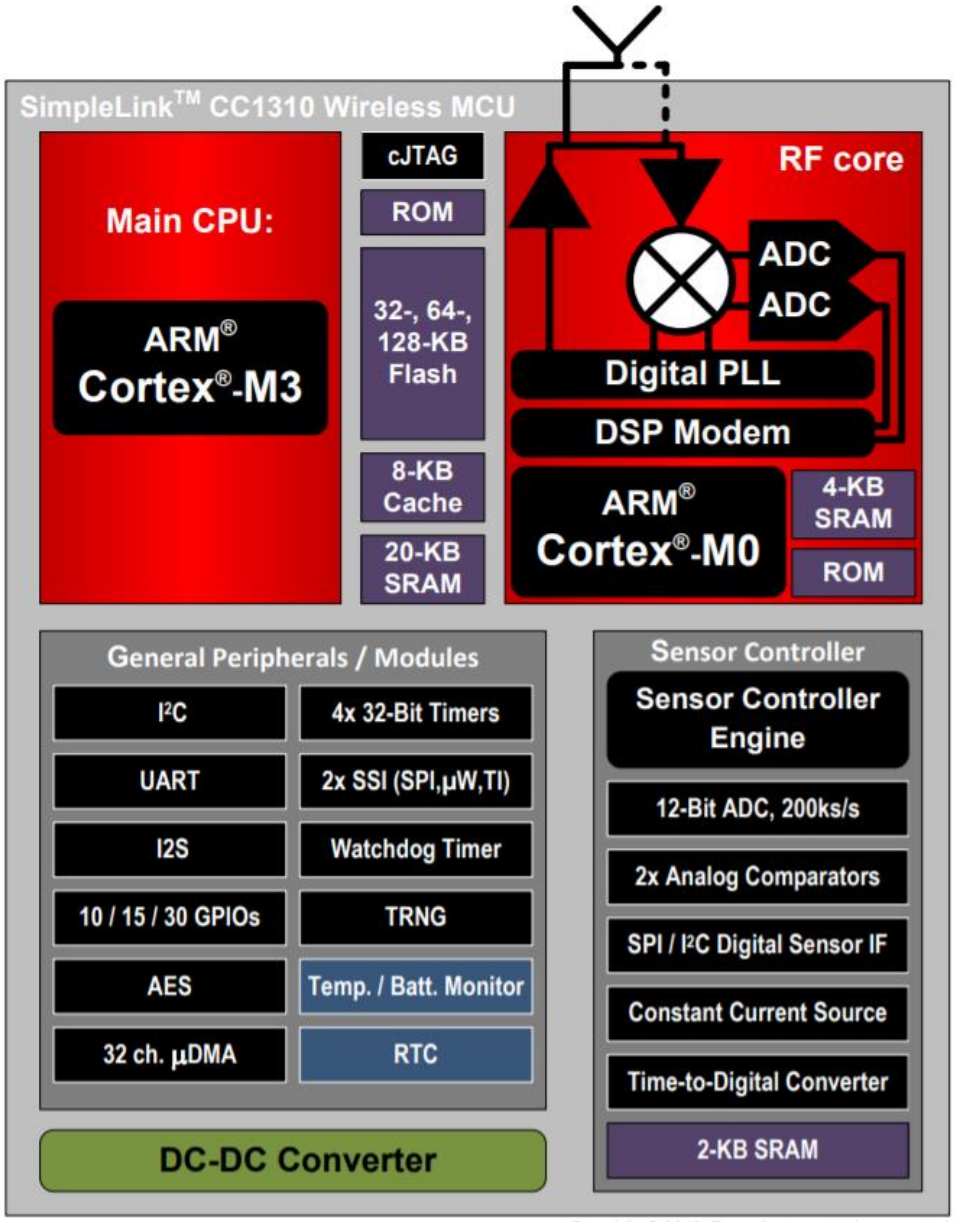


Figura 5-2. Diagrama de flujo del dispositivo CC1310.

5.3 CC1310 LAUNCHPAD.

La empresa desarrolladora del chip **CC1310 SimpleLink™ Ultra-Low-Power Sub-1 GHz Wireless MCU,** Texas Instruments, también desarrolló un launchpad con el chip integrado que abre diferentes posibilidades haciendo un dispositivo más completo y versátil.

Figura 5-1-1. CC1310 LaunchPad.

El dispositivo en cuestión, es parte de la plataforma de microcontroladores de Texas Instruments. Ofreciendo así un entorno de desarrollo flexible, con diversas opciones software que se enfocan a la comunicación cableada o inalámbrica, esta segunda es la interesante para este proyecto. El Launchpad ofrece una conexión de los conectores I/O que posee con una variedad de módulos de evaluación (EVM) y módulos plug-in Boosterpack[8].

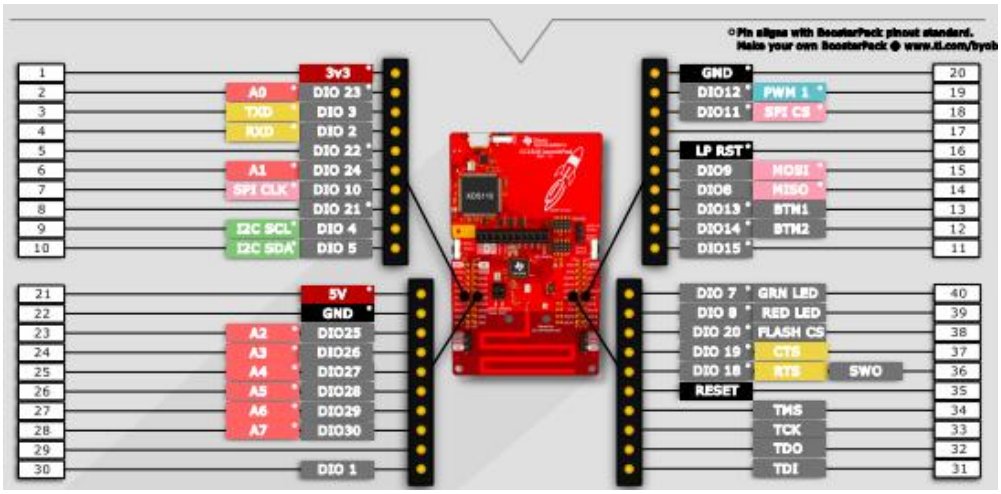


Figura 5-1-2. PINOUT CC1310 Launchpad.

En la imagen anterior vemos el PINOUT del Launchpad con sus respectivos pines de GND, 3v3 y 5 voltios. Además consta también de pines RX y TX para la comunicación serie y pines de PWM. El dispositivo posee una antena incorporada y una solución para la incorporación de otra antena más potente en caso de ser necesario. Por otro lado el dispositivo tiene 3 pulsadores, 1 de reset y dos programables que vienen preconfigurados con la interfaz serie del menú.

En este proyecto se usará el launchpad en vez de simplemente el dispositivo cc1310, ya que el primero desempeña la mayoría de las funciones por si mismo sin necesidad de desarrollar otra plataforma hardware auxiliar, además de que, los diferentes pines y módulos auxiliares abren un gran abanico de posibilidades a la hora de añadir funcionalidades nuevas al producto final.

5.4 Cumplimiento de requisitos.

Microcontrolador para el procesado de la información.	OK
Modo de bajo consumo.	OK
Modulo de seguridad.	OK
UART, I2C y RTC.	OK
Ancho de banda soportable para el protocolo	OK
Fabricante de la plataforma HW proporciona la pila de protocolos Wireless M-Bus	OK
Moderm compatible con las bandas de frecuencia Wireless M-Bus	OK
Moderm compatible con las modulaciones Wireless M-Bus	OK

El dispositivo elegido dispone de un microprocesador Cortex M3, el cual tiene hasta 128 kB de flash programable y una velocidad del reloj de hasta 48 MHz. Dispone también de puertos UART, Real Time Clock (RTC) e I2C.

Los anchos de banda del protocolo Wireless M-Bus, que varían en función del modo elegido para la comunicación de los nodos, son todos soportados por esta plataforma HW, ya que tiene integrado el “Wireless M-Bus (EN 13757-4)” y el “IEEE® 802.15.4g PHY”. Ambos necesarios para la realización de este trabajo de fin de grado.

Hemos mencionado antes la necesidad de un dispositivo seguro, por lo cual también era necesario un módulo de seguridad, el cual está integrado, siendo este el “AES-128 Security Module”.

Por último, mencionar el bajo consumo, necesario para el desarrollo de está aplicación duradera y robusta. En bajo consumo, el dispositivo consume 0.7 µA (RTC Running and RAM and CPU Retention), siendo necesario el despertado por un evento externo.

Como esta plataforma cumple todos los requisitos marcados por el SW para su correcto funcionamiento, se elige el LaunchPad CC1310 para el desarrollo del hub y la conexión de las diferentes etapas del envio y recepción de información.

6 DESARROLLO FIRMWARE

Para conseguir el objetivo de este proyecto, hemos de programar un firmware que comunique las diferentes placas en una configuración de red que tenga una placa “master” y múltiples “slaves” que le pasan información a la primera mediante comandos AT.

6.1 Prueba de funcionamiento de los módulos.

Al principio se procede realizando unas pequeñas pruebas iniciales, para comprobar el correcto funcionamiento de estas. Tras una serie de pruebas de envio de paquetes, se concluye que las placas son capaces de realizar su función correctamente. Pero se encuentra la problemática de que se ha sustituido el firmware original de la placa por los de las pruebas y se han desconfigurado ciertos parámetros. Para comprobar si la placa se ha corrompido se comprueba que información envía por puerto serie, donde la respuesta es nula.

Se procede a ir comprobando los mosulos más sencillos para ver si funcionan correctamente o no. Se comprueban los leds. Se programa un pequeño firmware que consiste en un pwm que enciende y apaga los leds cada 3 segundos. Comprobando que el funcionamiento en este caso es correcto.

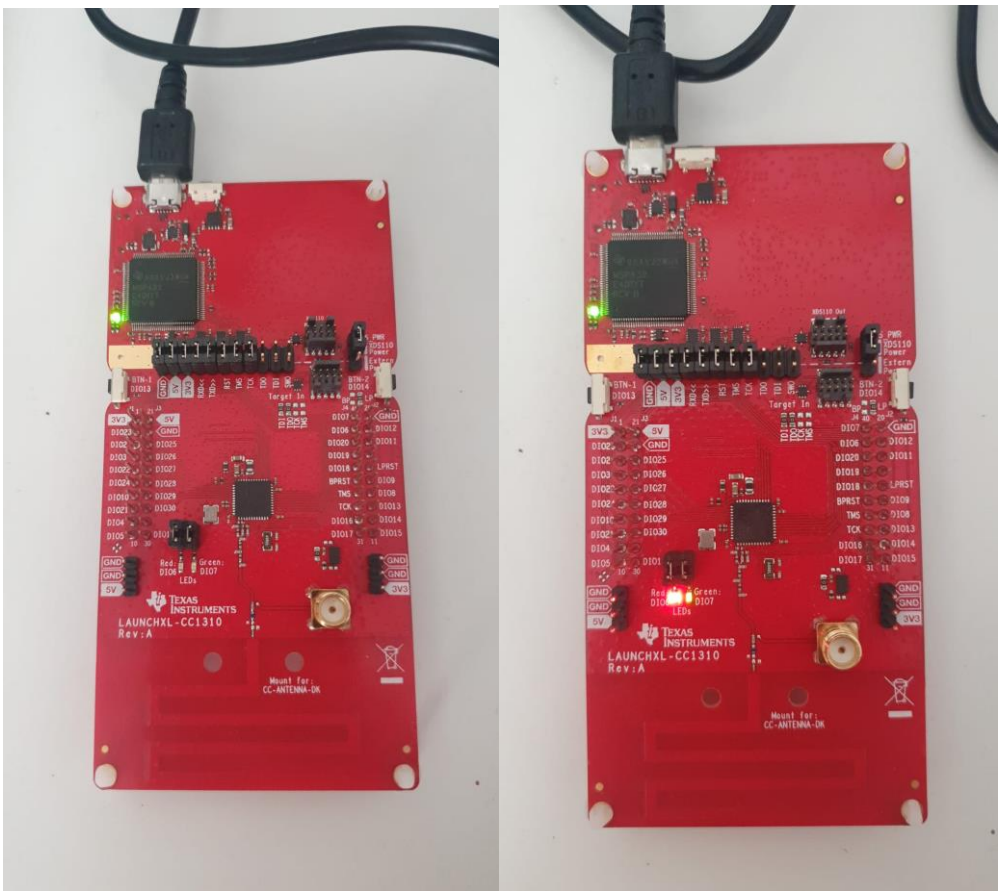


Figura 6-1. Comprobación funcionamiento leds DI06 y DI07.

6.2 Envio y recepción de datos de las placas en modo Master y Slave.

El siguiente paso es probar el envio y recepción de datos de las placas. Conectando los pines RX y TX de ambas placas y con un pequeño firmware que consiste en el envio de una cadena de caracteres a través de la UART. Dicha cadena de caracteres se envia por puerto serie y pasa del TX de una de las placas al RX de la otra, produciendo un eco que se envía por el TX de la segunda placa al RX de la primera. La cadena de caracteres se puede leer a través de un monitor serie.

El código de prueba del eco se puede ver a continuación.

```
#include <stdint.h>
#include <stddef.h>

/* Driver Header files */
#include <ti/drivers/GPIO.h>
#include <ti/drivers/UART.h>
#include "Board.h"

void *mainThread(void *arg0)
{
    char        input;
    const char  echoPrompt[] = "cadenaejemplo\r\n";
```

```
    UART_Handle uart;
    UART_Params uartParams;

    /* Call driver init functions */
    GPIO_init();
    UART_init();

    /* Configure the LED pin */
    GPIO_setConfig(Board_GPIO_LED0, GPIO_CFG_OUT_STD | GPIO_CFG_OUT_LOW);

    /* Turn on user LED */
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_ON);

    /* Create a UART with data processing off. */
    UART_Params_init(&uartParams);
    uartParams.writeDataMode = UART_DATA_BINARY;
    uartParams.readDataMode = UART_DATA_BINARY;
    uartParams.readReturnMode = UART_RETURN_FULL;
    uartParams.readEcho = UART_ECHO_OFF;
    uartParams.baudRate = 115200;

    uart = UART_open(Board_UART0, &uartParams);

    if (uart == NULL) {
        /* UART_open() failed */
        while (1);
    }

    UART_write(uart, echoPrompt, sizeof(echoPrompt));

    /* Bucle de eco */
    while (1) {
        UART_read(uart, &input, 1);
        UART_write(uart, &input, 1);
    }
}
```

6.3 Envío de datos inalámbricos.

El siguiente paso consiste en probar la conexión inalámbrica de ambos LaunchPads. Al igual que en el caso anterior, uno tiene que ser el transmisor(TX) y el otro el receptor(RX). En este ejemplo se usa EasyLink, un protocolo de el envio de datos por radiofrecuencia que se usa en componentes hardware de Texas Instruments™.

Este ejemplo consiste en el envio de un paquete por parte de una de las placas(TX), señalizando el envio del paquete con el encendido de un led de color verde. Y la recepción de dicho paquete por parte de la otra placa(RX), señalizando dicha recepción con el encendido del led rojo. Como se puede observar en la figura 5-3, ambos leds se encienden de forma simultanea, indicando el correcto envio y recepción de paquetes, y después de un breve parpadeo, ambos se apagan.

Por el contrario si la placa receptora no recibe paquetes, se queda en un estado de espera encenciendo alternativamente ambos leds como se puede observar en la siguiente imagen. Este encendido alternativo significa que, cuando el led rojo está encendido la placa está esperando recibir paquetes, y como no los obtiene, enciende el led verde para indicarnos que no ha recibido ninguno. Por el contrario si la placa transmisora envía paquetes pero ningún otro nodo los recibe, esto se manifiesta por el apagado de ambos leds de usuario indicando que los paquetes que se están enviando, no están siendo recibidos por otro punto de la red.

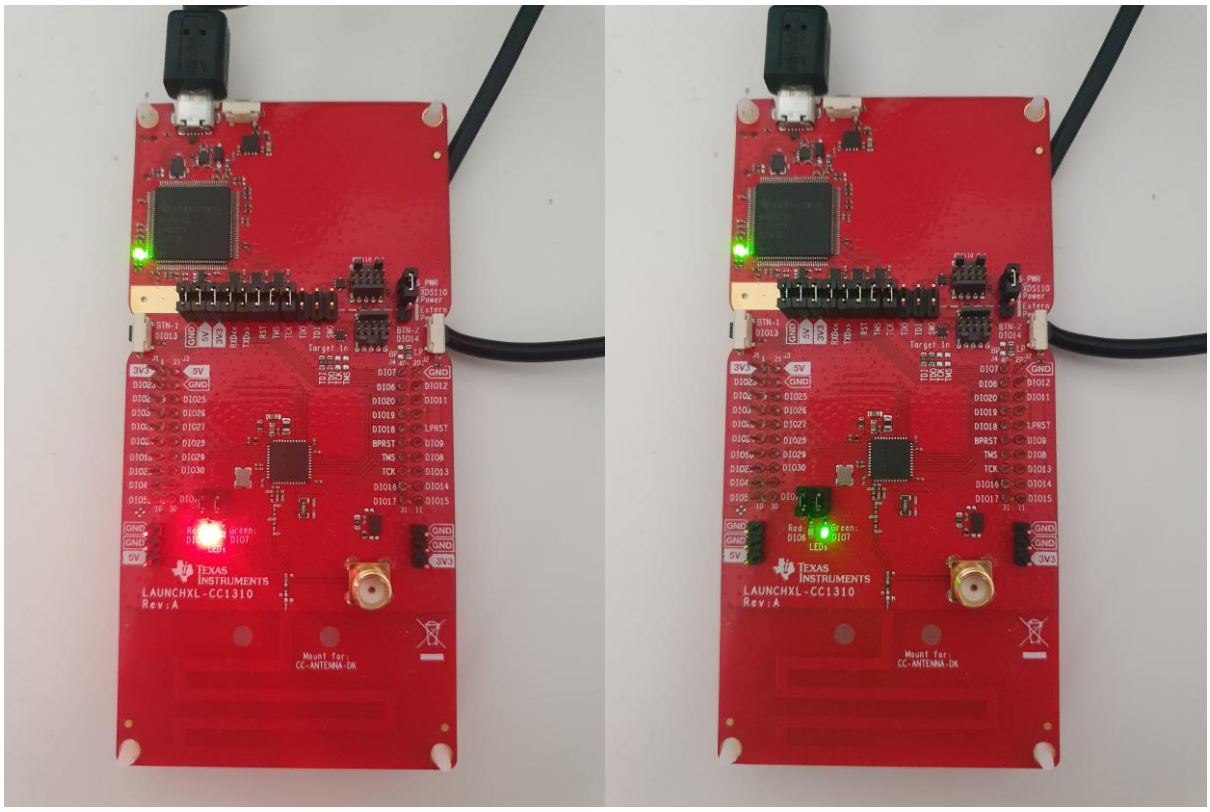


Figura 6-2. Estado de espera de paquetes placa RX.

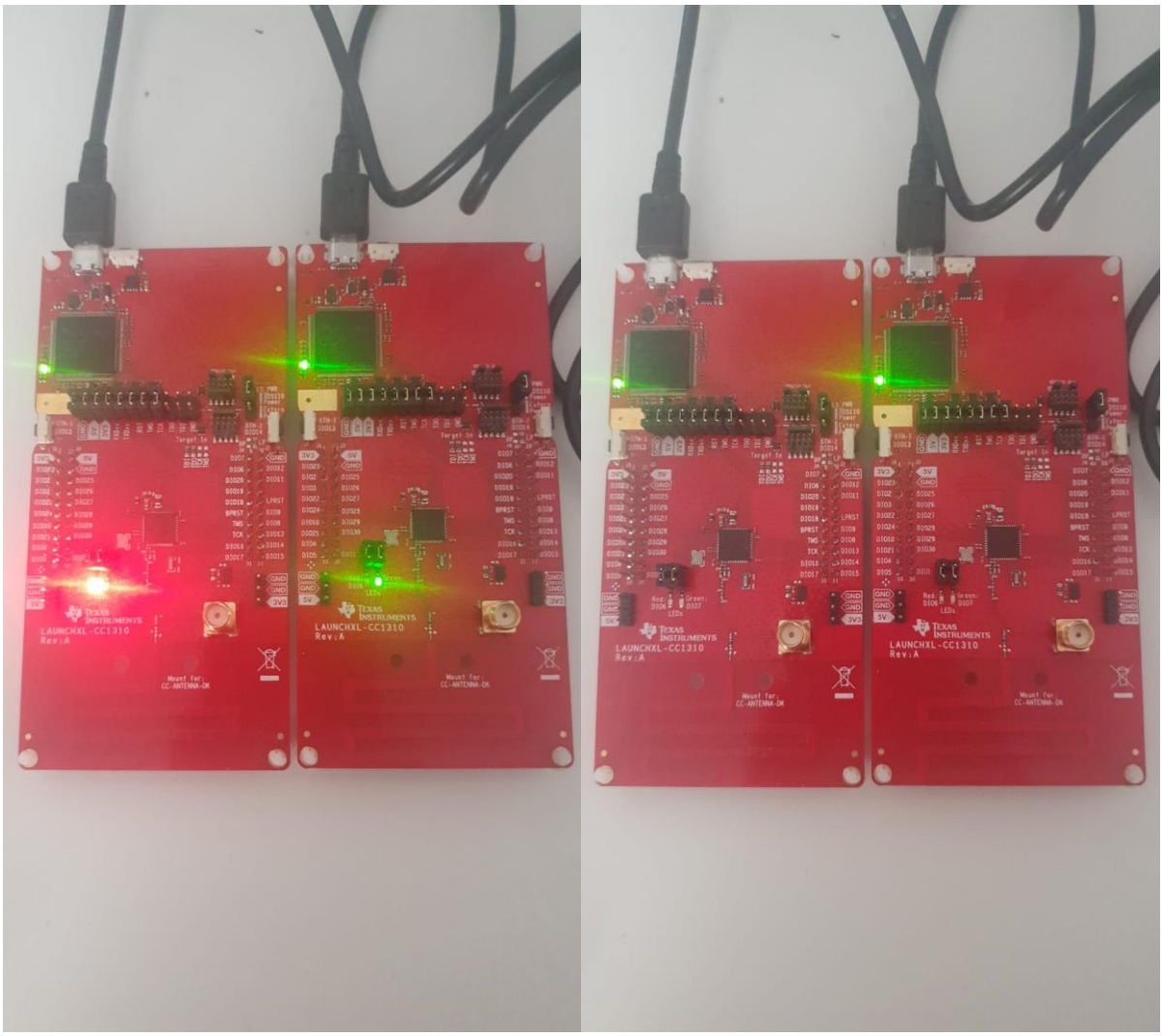


Figura 6-3. Envio y recepción de paquetes.

El código utilizado para el envio es el siguiente.

```
#include <stdlib.h>

/* XDCtools Header files */
#include <xdc/std.h>
#include <xdc/runtime/Assert.h>
#include <xdc/runtime/Error.h>
#include <xdc/runtime/System.h>

/* BIOS Header files */
#include <ti/sysbios/BIOS.h>
#include <ti/sysbios/knl/Task.h>
#include <ti/sysbios/knl/Semaphore.h>
#include <ti/sysbios/knl/Clock.h>

/* TI-RTOS Header files */
#include <ti/drivers/PIN.h>

/* Board Header files */
#include "Board.h"

/* Application Header files */
#include "smartrf_settings/smartrf_settings.h"
```

```
/* EasyLink API Header files */
#include "easylink/EasyLink.h"

/* Undefine to not use async mode */
#define RFEASYLINKECHO_ASYNC

#define RFEASYLINKECHO_TASK_STACK_SIZE    1024
#define RFEASYLINKECHO_TASK_PRIORITY      2

#define RFEASYLINKECHO_PAYLOAD_LENGTH     30

Task_Struct echoTask;    /* not static so you can see in ROV */
static Task_Params echoTaskParams;
static uint8_t echoTaskStack[RFEASYLINKECHO_TASK_STACK_SIZE];

/* Pin driver handle */
static PIN_Handle pinHandle;
static PIN_State pinState;

/*
 * Application LED pin configuration table:
 *   - All LEDs board LEDs are off.
 */
PIN_Config pinTable[] = {
    Board_PIN_LED1 | PIN_GPIO_OUTPUT_EN | PIN_GPIO_LOW | PIN_PUSHPULL |
PIN_DRVSTR_MAX,
    Board_PIN_LED2 | PIN_GPIO_OUTPUT_EN | PIN_GPIO_LOW | PIN_PUSHPULL |
PIN_DRVSTR_MAX,
    PIN_TERMINATE
};

static uint16_t seqNumber;

#ifdef RFEASYLINKECHO_ASYNC
static Semaphore_Handle echoDoneSem;
#endif //RFEASYLINKECHO_ASYNC

EasyLink_TxPacket txPacket = {{0}, 0, 0, {0}};

bool isPacketCorrect(EasyLink_RxPacket *rxp, EasyLink_TxPacket *txp)
{
    uint16_t i;
    bool status = true;

    for(i = 0; i < rxp->len; i++)
    {
        if(rxp->payload[i] != txp->payload[i])
        {
            status = false;
            break;
        }
    }
    return(status);
}

#ifdef RFEASYLINKECHO_ASYNC
void echoTxDoneCb(EasyLink_Status status)
{
    if (status == EasyLink_Status_Success)
```

```
    {
        /* Toggle LED1 to indicate TX */
        PIN_setOutputValue(pinHandle,
Board_PIN_LED1,!PIN_getOutputValue(Board_PIN_LED1));
        /* Turn LED2 off, in case there was a prior error */
        PIN_setOutputValue(pinHandle, Board_PIN_LED2, 0);
    }
    else
    {
        /* Set both LED1 and LED2 to indicate error */
        PIN_setOutputValue(pinHandle, Board_PIN_LED1, 1);
        PIN_setOutputValue(pinHandle, Board_PIN_LED2, 1);
    }

    Semaphore_post(echoDoneSem);
}

void echoRxDoneCb(EasyLink_RxPacket * rxPacket, EasyLink_Status status)
{
    if ((status == EasyLink_Status_Success) &&
            (isPacketCorrect(rxPacket, &txPacket)))
    {
        /* Toggle LED1, clear LED2 to indicate Echo RX */
        PIN_setOutputValue(pinHandle,
Board_PIN_LED1,!PIN_getOutputValue(Board_PIN_LED1));
        PIN_setOutputValue(pinHandle, Board_PIN_LED2, 0);
    }
    else if (status == EasyLink_Status_Aborted)
    {
        /* Set LED2 and clear LED1 to indicate Abort */
        PIN_setOutputValue(pinHandle, Board_PIN_LED2, 1);
        PIN_setOutputValue(pinHandle, Board_PIN_LED1, 0);
    }
    else
    {
        /* Set both LED1 and LED2 to indicate error */
        PIN_setOutputValue(pinHandle, Board_PIN_LED1, 1);
        PIN_setOutputValue(pinHandle, Board_PIN_LED2, 1);
    }

    Semaphore_post(echoDoneSem);
}
#endif //RFEASYLINKECHO_ASYNC

static void rfEasyLinkEchoTxFnx(UArg arg0, UArg arg1)
{
    uint32_t absTime;

#ifdef RFEASYLINKECHO_ASYNC
    /* Create a semaphore for Async */
    Semaphore_Params params;
    Error_Block      eb;

    /* Init params */
    Semaphore_Params_init(&params);
    Error_init(&eb);

    /* Create semaphore instance */
    echoDoneSem = Semaphore_create(0, &params, &eb);
    if(echoDoneSem == NULL)
```

```
    {
        System_abort("Semaphore creation failed");
    }

#else
    EasyLink_RxPacket rxPacket = {{0}, 0, 0, 0, 0, {0}};
#endif //RFEASYLINKECHO_ASYNC

    // Initialize the EasyLink parameters to their default values
    EasyLink_Params easyLink_params;
    EasyLink_Params_init(&easyLink_params);

    /*
     * Initialize EasyLink with the settings found in easylink_config.h
     * Modify EASYLINK_PARAM_CONFIG in easylink_config.h to change the default
     * PHY
     */
    if(EasyLink_init(&easyLink_params) != EasyLink_Status_Success)
    {
        System_abort("EasyLink_init failed");
    }

    /*
     * If you wish to use a frequency other than the default, use
     * the following API:
     * EasyLink_setFrequency(868000000);
     */

    // Packet Originator
    while(1) {
        /* Create packet with incrementing sequence number and random payload */
        txPacket.payload[0] = (uint8_t)(seqNumber >> 8);
        txPacket.payload[1] = (uint8_t)(seqNumber++);
        uint8_t i;
        for (i = 2; i < RFEASYLINKECHO_PAYLOAD_LENGTH; i++)
        {
            txPacket.payload[i] = rand();
        }

        txPacket.len = RFEASYLINKECHO_PAYLOAD_LENGTH;

        /*
         * Address filtering is enabled by default on the Rx device with the
         * an address of 0xAA. This device must set the dstAddr accordingly.
         */
        txPacket.dstAddr[0] = 0xaa;

        /* Set Tx absolute time to current time + 1000ms */
        if(EasyLink_getAbsTime(&absTime) != EasyLink_Status_Success)
        {
            // Problem getting absolute time
        }
        txPacket.absTime = absTime + EasyLink_ms_To_RadioTime(1000);

#ifdef RFEASYLINKECHO_ASYNC
        EasyLink_transmitAsync(&txPacket, echoTxDoneCb);

        /* Wait for Tx to complete. A Successful TX will cause the echoTxDoneCb
         * to be called and the echoDoneSem to be released, so we must
         * consume the echoDoneSem
```

```
         */
        Semaphore_pend(echoDoneSem, BIOS_WAIT_FOREVER);

        /* Switch to Receiver */
        EasyLink_receiveAsync(echoRxDoneCb, 0);

        /* Wait 500ms for Rx */
        if(Semaphore_pend(echoDoneSem, (500000 / Clock_tickPeriod)) == FALSE)
        {
            /* RX timed out abort */
            if(EasyLink_abort() == EasyLink_Status_Success)
            {
               /* Wait for the abort */
               Semaphore_pend(echoDoneSem, BIOS_WAIT_FOREVER);
            }
        }
#else
        EasyLink_Status result = EasyLink_transmit(&txPacket);

        if (result == EasyLink_Status_Success)
        {
            /* Toggle LED1 to indicate TX */
            PIN_setOutputValue(pinHandle,
Board_PIN_LED1,!PIN_getOutputValue(Board_PIN_LED1));
            /* Turn LED2 off, in case there was a prior error */
            PIN_setOutputValue(pinHandle, Board_PIN_LED2, 0);
        }
        else
        {
            /* Set both LED1 and LED2 to indicate error */
            PIN_setOutputValue(pinHandle, Board_PIN_LED1, 1);
            PIN_setOutputValue(pinHandle, Board_PIN_LED2, 1);
        }

        /* Switch to Receiver, set a timeout interval of 500ms */
        rxPacket.absTime = 0;
        rxPacket.rxTimeout = EasyLink_ms_To_RadioTime(500);
        result = EasyLink_receive(&rxPacket);

        /* Check Received packet against what was sent, it should be identical
         * to the transmitted packet
         */
        if (result == EasyLink_Status_Success &&
                isPacketCorrect(&rxPacket, &txPacket))
        {
            /* Toggle LED1, clear LED2 to indicate Echo RX */
            PIN_setOutputValue(pinHandle,
Board_PIN_LED1,!PIN_getOutputValue(Board_PIN_LED1));
            PIN_setOutputValue(pinHandle, Board_PIN_LED2, 0);
        }
        else if (result == EasyLink_Status_Rx_Timeout)
        {
            /* Set LED2 and clear LED1 to indicate Rx Timeout */
            PIN_setOutputValue(pinHandle, Board_PIN_LED2, 1);
            PIN_setOutputValue(pinHandle, Board_PIN_LED1, 0);
        }
        else
        {
            /* Set both LED1 and LED2 to indicate error */
            PIN_setOutputValue(pinHandle, Board_PIN_LED1, 1);
```

```
            PIN_setOutputValue(pinHandle, Board_PIN_LED2, 1);
        }
#endif //RFEASYLINKECHO_ASYNC
    }
}

void echoTask_init(PIN_Handle inPinHandle) {
    pinHandle = inPinHandle;

    Task_Params_init(&echoTaskParams);
    echoTaskParams.stackSize = RFEASYLINKECHO_TASK_STACK_SIZE;
    echoTaskParams.priority = RFEASYLINKECHO_TASK_PRIORITY;
    echoTaskParams.stack = &echoTaskStack;
    echoTaskParams.arg0 = (UInt)1000000;

    Task_construct(&echoTask, rfEasyLinkEchoTxFnx, &echoTaskParams, NULL);
}

/*
 *  ======== main ========
 */
int main(void)
{
    /* Call driver init functions. */
    Board_initGeneral();

    /* Open LED pins */
    pinHandle = PIN_open(&pinState, pinTable);
    Assert_isTrue(pinHandle != NULL, NULL);

    /* Clear LED pins */
    PIN_setOutputValue(pinHandle, Board_PIN_LED1, 0);
    PIN_setOutputValue(pinHandle, Board_PIN_LED2, 0);

    echoTask_init(pinHandle);

    /* Start BIOS */
    BIOS_start();

    return (0);
}
```

El código de recepción de paquetes es similar, sustitute la escritura y el envio por las funciones de lectura y recepción, y modifica el encendido de leds en caso de no recibir de forma correcta el paquete. No se reflejará aquí para no poner dos códigos muy parecidos.

6.4 Explicación del “Wireless M-Bus Software Stack”.

Ahora nos metemos directamente con el SW Stack de STACKFORCE para Wireless M-Bus. Primero se va a explicar de forma breve las funciones principales a usar para ambos tipos de nodos y alguna función de la interfaz necesaria para la obtención de datos de los dispositivos utilizados[9].

Se mencionarán primero las funciones de interfaz.

-wmbus_apl_cleanUp(); Limpia la memoria.

-wmbus_apl_setDeviceAddr(); Establece la dirección de Wireless M-Bus.

-wmbus_apl_getDeviceAddr(); Obtiene la dirección de Wireless M-Bus.

Ahora pasamos a las funciones propias del nodo “meter”.

-wmbus_apl_mtr_init(); Inicializa el dispositivo “meter”.

-wmbus_apl_mtr_run(); Comienza el funcionamiento el dispositivo “meter”.

-wmbus_apl_mtr_send(); Envio de datos.

-wmbus_apl_mtr_sendInstallationRequest(); Comienza el proceso de instalación.

-wmbus_apl_mtr_setCollectorAdress(); Marca la dirección Wireless M-Bus del colector.

-wmbus_apl_mtr_getCollectorAdress(); Obtiene la dirección Wireless M-Bus del colector.

-wmbus_apl_mtr_setUserData(); Marca como va a ser el formato de los datos a enviar.

-wmbus_apl_mtr_indication_rx(); Avisa de un dato listo para el envío.

Por último mencionamos las funciones propias del nodo “collector”.

-wmbus_apl_col_init(); Inicializa el dispositivo “collector”.

-wmbus_apl_col_run(); Comienza el funcionamiento el dispositivo “collector”.

-wmbus_apl_col_addMeter(); Añade un nuevo “meter” a la lista.

-wmbus_apl_col_removeMeter(); Quita un “meter” de la lista.

-wmbus_apl_col_getMeterAddr(); Obtiene la dirección Wireless M-Bus del meter.

-wmbus_apl_col_indication_rx(); Se activa si un nuevo dato está esperando para ser leido.

-wmbus_apl_col_indication_newMeter(); Un nuevo nodo “meter” solicita conexión.

Se pretende programar un código simple para el envío y la recepción de datos, de forma que se tengan como parámetros seleccionables los diferentes modos de funcionamiento S, T, C y N. También se pretende saber en todo momento el número de contadores que hay conectado a un mismo nodo “collector” y a su vez, poder seleccionar de qué nodo se ven los datos.

6.5 Comandos utilizados.

Para la programación de este protocolo de envio de datos inalámbrico, se usan una serie de comandos propios de dicho protocolo. Al igual que Quectel usa los comandos AT para todas las funcionalidades del dispositivo como el reinicio, el despertado o el envio de tramas, el protocolo Wireless M-Bus tiene su propio set de comandos, los cuales tienen un identificador que varía en base a la funcionalidad que se quiere implementar. Yendo este identificador de 00 a 0B e implementando las funcionalidades como puede verse en la siguiente tabla.[9]

Code	Command name	Page
00	Communication mode	12
01	Link address	13
02	Application address	14
03	AES key	15
04	Access number	15
05	Status byte	16
06	Contents of meter telegram	17
07	Transmit mode	18
08	Transmit power	19
09	Module information	19
0A	Sleep mode control	20
0B	Set MUC flags	20

Dichos parámetros se usan para la configuración del módulo. Todos los comandos tienen la misma nomenclatura, empezando todos con el caracter '>' indicando el envío. Por el contrario, todas las respuestas recibidas de estos comandos comenzarán con el carácter '<', lo que hace más entendible y mejor interpretable el envio de comando y la recepción de respuestas. Todos los comandos y respuestas van con el terminador [CR] y todos los paquetes enviados han de ir precedidos con el carácter que indica el "wake up" 0x00 y una pausa de 2 ms.

Por lo que un envio de comando siempre será de este modo: >[cc] [rw][data][CR]. Donde "cc" corresponde con el código de comando visto en la tabla anterior, el "rw" corresponde con el modo de lectura o escritura, indicando un '?' si será de lectura, o un ':' si escritura. Y por último "data" que serán los datos adicionales del comando.

Una respuesta genérica a un envío genérico del comando como el anterior, será: <[data][CR]. Donde "CR" es el terminador genérico y "data" la respuesta correspondiente al envio. Esta respuesta puede variar entre tres posibilidades:

- <OK[CR] Para indicar la recepción correcta del comando.
- <ERR1[CR] Para indicar un error de sintaxis.
- <ERR2[CR] Para indicar un valor incorrecto de los parámetros de entrada.

Los comandos utilizan una configuración de las palabras que contiene información importante sobre la condición concreta del dispositivo Wireless M-Bus. Los bits 2 y 3 pueden ser modificados por la orden "06" que modifica el contenido del datagrama, y el bit 13 puede ser modificado por la orden 7 que define el modo síncrono o asíncrono.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Bidirectional communication	Accessibility	Synchronous	Reserved	Mode bit3	Mode bit2	Mode bit1	Mode bit0	Number of encrypted blocks	Number of encrypted blocks	Number of encrypted blocks	Number of encrypted blocks	Content of telegram	Content of telegram	Hop counter	Hop counter

El primer comando, con el identificador "00" trata del establecimiento del modo de funcionamiento del protocolo. Como se vio anteriormente, existen 4 modos de funcionamiento diferentes, el C, S, T y N. Los modos C y N, por sus características, se han de programar a parte y usar posteriormente uno de los modos de comunicación que vamos a ver a continuación. Mientras que los modos T y S se configurarán de la siguiente forma.

Input value	Mode
10	Meter - S1
11	Meter - T1
12	Meter - S2
13	Meter - T2
22	MUC - S2 (test)
23	MUC - T2 (test)

Tal cual hemos visto antes el envio de comandos, para establecer uno de estos modos, enviaremos el comando como: >00:10[CR], este el '>' indica el envio del comando, el "00" es el identificador de establecimiento de modo, el "10" indica que queremos configurar el meter con el modo de envio S1, el cual se diferencia de S2 porque este último es bidireccional mientras que S1 es unidireccional. Lo mismo ocurre con los modos T1 y T2.

Los valores de entrada del 10 al 13 se corresponden con modos de configuración de los dispositivos Meter, mientras que el 22 y 23 corresponden con los modos de configuración de los nodos Collector. Este comando si se realiza correctamente ha de devolver un <OK[CR] indicando el establecimiento correcto del modo de funcionamiento.

El segundo comando corresponde con el identificador 01, que conecta los nodos meter y collector y reserva el espacio necesario en memoria para guardar la información de este dispositivo. Este comando tiene que incluir diferentes identificadores como se podrá ver a continuación. >01:[MMMM][IIIIIIII][VV][DD][CR]. Donde:

- [MMMM]: Corresponde con el Manufacturer ID. 2 bytes en hexadecimal.
- [IIIIIIII]: Corresponde con el serial numer del meter a agregar. 4 bytes en hex.
- [VV]: Es el número de versión del meter. 1 byte en hexadecimal.
- [DD]: El código del dispositivo a agregar a la lista. 1 byte en hexadecimal.

Un ejemplo de funcionamiento de dicho comando puede ser el siguiente:

>01: 5336443322111037 [CR]. Donde el 01 corresponde con el identificador de conexión del meter. Los primeros cuatro números "3653" corresponden con el Manufacturer ID, los siguientes 8 "11223344" equivaldrían al serial number del meter, el "10" corresponde con el número de versión de dicho meter y por último el "37" final correspondería con un identificador que marca el tipo de dispositivo.

El comando, si se ha realizado correctamente nos devolverá <OK[CR]. Indicando el que el meter indicado en los parámetros se han añadido a la lista del collector.

También podemos usar el comando de otra manera. Si lo formulamos de la siguiente forma, >01?[CR], Estamos preguntándole al collector los meter asociados a el. Donde la respuesta será, por ejemplo: <abcd67452301b1cf. Siendo el Manufacturer ID "cdab", el serial number "0123456", el número de versión "b1" y el código del dispositivo será "cf".

El tercer comando correspondiente con el identificador 02, tiene los mismos parámetros de entrada que el anterior, siendo este formulado de la siguiente manera. >02:[MMMM][IIIIIIII][VV][DD][CR]. La diferencia de estos radica en el funcionamiento. Este comando se usa para seleccionar de cual de todos los meter conectados vamos a recibir la información. La introducción del comando es la misma y al igual que antes, y si lo introducimos como >02?[CR], nos dirá el meter al que se esta asociando la información recibida.

El cuarto comando, que tiene el identificador 03, introduce una contraseña al collector, el cual nos la pedirá para poder acceder a la información que está recibiendo. El comando se formula de la siguiente manera. >03:[KKKKKKKKKKKKKKKKKKKKKKKKKKKKKK][CR]. Donde el 03 corresponde con el identificador de establecimiento de contraseña, y el [KKKKKKKKKKKKKKKKKKKKKKKKKKKKKK] corresponde con la contraseña a

introducir, que consiste de 16 bytes en hexadecimal.

Un ejemplo de funcionamiento de dicho comando puede ser el mostrado a continuación. >03:0102030405060708090a0b0c0d0e0f[CR]. Siendo la contraseña los números del 1 al 15 expresados en hexadecimal. La respuesta, al igual que los comandos anteriores, es un <OK[CR].

También se puede usar el comando como >03?[CR], donde se pedirá la contraseña establecida actualmente. En el ejemplo anterior, la respuesta recibida de este comando sería <0102030405060708090a0b0c0d0e0f[CR].

El quinto comando corresponde con el identificador 04, y sirve para establecer el Wireless M-Bus Access number. El cual, combinado con la dirección de transmisión nos servirá para identificar el datagrama enviado. Este numero consta de 1 byte en hexadecimal y puede establecerse a través de este comando, o será generado automáticamente.

El módulo transmisor incrementará este numero en uno por cada transmisión síncrona que se realice. Para las transmisiones asíncronas, se utilizará el Access number de la última transmisión síncrona realizada.

Un ejemplo de uso de este comando puede ser >04:03. Donde se establecerá el Access number del meter seleccionado a 3. Al igual que en los anteriores comandos, la respuesta deseada será <OK[CR].

Y también se puede preguntar el Access number del meter seleccionado a través del uso del comando >04?[CR]. Donde la respuesta recibida será <03[CR] en el caso del ejemplo anterior.

El sexto comando correspondiente con el identificador 05, establece el byte de "status" de la conexión. Este byte sirve como identificador del datagrama y el comando se ejecuta como los anteriores. >05:[SS][CR]. Siendo SS el byte en hexadecimal. Un ejemplo de funcionamiento puede ser >05:1b[CR]. Donde se establece el byte de status a "1b".

El séptimo comando correspondiente con el identificador 06, establece el contenido del datagrama que se va a enviar. Este comando uno de los mas importantes para establecer el envio de la información de forma correcta.

El comando se ejecuta como el resto >06:[CC][CR]. Donde CC es un identificador del telegrama enviado por el meter. Este puede tomar 4 valores diferentes:

- **00**: Telegrama estándar con una variable sin signo proveniente del meter.
- **04**: Telegrama de datos con signo (consta de datos del meterr con una firma aprobada para facturación)
- **08**: Telegrama estático. Con parámetros y definiciones que no suelen cambiar.
- **0c**. Reservado, pero no implementado.

Al igual que en los otros comandos, se puede preguntar que modo de contenido de telegramas está establecido con >06?[CR]. Donde se recibirá una respuesta como <04[CR].

El octavo comando se corresponde con el identificador 07. Este comando establece si el envio de datagramas se realiza de forma síncrona o asíncrona. >07:[MM][CR], donde el identificador MM puede tomar dos valores diferentes: 00 para establecer el envio de datos de forma asíncrona, o 20 para establecerlo de forma síncrona.

Se puede preguntar al MCU qué modo de envio de datos está establecido con >07?[CR]. El cual nos devolverá un <20[CR] si es síncrono, o un <10[CR] si es asíncrono.

El noveno comando identificado con 08, establece la potencia de la señal que se está enviando. El comando se envia con el >08:[PP][CR]. Donde el identificador PP corresponde a un byte en hexadecimal que indica la potencia establecida. Esta puede variar entre las siguientes:

- 00: -30 dBm.
- 01: -24 dBm.
- 02: -12 dBm.
- 03: -6 dBm.
- 04: 0 dBm.
- 05: +5 dBm.
- 06: +10 dBm.
- 07: +12 dBm.

Se puede pedir al MCU, al igual que en los anteriores comandos, que nos indique qué potencia de señal tenemos establecida ahora mismo, con >08?[CR].

El décimo comando correspondiente con el identificador 09, nos indica la versión del HardWare de la PCB de la placa y el FirmWare cargado en ella. Se usa de la siguiente manera: >09?[CR]. Donde recibiremos una respuesta en el formato <[HW][FW][CR] con dos decimales de la versión. Un ejemplo podría ser <1.061.23[CR]. Donde la versión de la PCB HW será la 1.06 y la versión del FW cargado en la placa será la 1.23.

El undécimo comando se corresponde con el identificador 0A. Este comando nos sirve para dormir, despertar y resetear el módulo. Tiene un identificador para elegir entre las tres opciones que este módulo nos ofrece.

La ejecución de este comando se hará como se muestra a continuación: >0A: [SM][CR]. Donde SM que corresponde con las siglas de Sleep Mode, será el indentificador donde expresaremos qué acción tiene que realizar el dispositivo. Este identificador puede tomar los siguientes valores:

- **00:** Si queremos activar el modo Deep Sleep.
- **01:** Si queremos despertar el dispositivo y ponerlo a la escucha.
- **02:** Si queremos hacerle un “reset” al dispositivo.

Unos ejemplos de funcionamiento pueden ser los siguientes:

- >0A:00[CR]. Pone a dormir el dispositivo.
- >0A:01[CR]. Solicitud de despertado del dispositivo.
- >0A:02[CR]. Reset inmediato del dispositivo.

Si ejecutamos el comando >0A?[CR], el dispositivo nos informará si se encuentra en modo Deep Sleep o en escucha.

El duodécimo comando corresponde con el identificador 0B se usa para activar o desactivar las diferentes banderas de el nodo collector.

El decimotercer comando se usa para comprobar el nivel de la batería. Corresponde con el identificador 0C. Se usa de la siguiente forma >0C?[CR]. La respuesta de este comando será <[BS][CR], donde BS corresponde con el battery status. Este puede tomar dos valores, 0x00 si la el voltaje de alimentación es correcto, o 0xff si la alimentación es demasiado baja.

El decimocuarto comando corresponde con el identificador 0D, y sirve para comprobar el serial number del dispositivo. Se usa de la siguiente forma: >0D?[CR], y nos devuelve la respuesta

en el siguiente formato. <[NNNNNN][CR] donde NNNNNN es el numero de 6 bytes en hexadecimal indicando el serial number del dispositivo meter o collector.

Los datos listos para enviar por Wireless M-Bus están contenidos en un paquete que tiene el siguiente formato [head][LEN][CMD][data][CRC]. Estos campos significan lo siguiente:

- Head: El primer byte siempre tiene el valor de 0xff.
- LEN: Longitud total de bytes después del campo LEN incluyendo el "checksum"
- CMD: Byte de identificación del comando.
- Data: Datos listos para enviar.
- CRC: "OR" exclusivo de todos los bytes después de la cabecera y previo al checksum

La respuesta al envio de este mensaje tiene el siguiente formato [LEN][result][data][CRC], donde LEN y CRC tienen el mismo valor que en el envio de datos, y "result" significa el resultado del envio del comando, y "data" los datos recibidos de vuelta del comando.

Un ejemplo de envio de datos se puede ver reflejado de la siguiente manera.

0x1580447A00000102030405060708090a0b0c0d0e0f23

- 0xff será el byte de cabecera
- LEN = 0x15 ([CMD]+[data]+[CRC]=21 bytes)
- CMD = 0x80
- CF = declara el tipo de mensaje acorde con la especificación OMS y en este caso tiene el valor 0x44 (SND-NR)
- CIF = función del telegrama y el tipo de codificación 0x7A
- AES-BC = 0x00 y es el número de bloques encriptados
- User data = 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
- CRC = 0x23

El modulo procesa la información recibida en el datagrama y envia una respuesta acorde, la cual en este ejemplo será **0x020002**.

- LEN = 0x02
- Result = 0x00, mensaje enviado de forma correcta
- Data = null, respuesta sin datos
- CRC = 0x02.

Con esto se dejan explicado el funcionamiento de los comandos necesarios para el procesamiento de datos y la configuración del módulo. Debemos destacar los comandos de selección de modo (S, T, C o N) y el comando de Deep Sleep, que serán los necesarios para hacer una aplicación de bajo consumo que nos permita la correcta transmisión de datos de varios dispositivos meter a un collector.

En conclusión, vamos a ver un pequeño resumen sobre los comandos a utilizar y un pequeño ejemplo de uso, para tener más recogido el uso de estos comandos y las diferentes funcionalidades que estos ofrecen.

En la siguiente tabla se muestran los diferentes comandos, su identificador, la función

implementada y un pequeño ejemplo de funcionamiento, con el cual se entenderá mejor el modo de uso.

Identificador.	Comando.	Función y ejemplo.
00	Communication Mode	Selecciona el modo de comunicación entre S, C, T y N. Ej: >00:10[CR] //set meter S1 mode || >00:13[CR] //set meter T2 mode. (Según la tabla de identificadores asociada a este comando visto anteriormente)
01	Link Address	Registra un meter en la lista del collector y reserva la memoria. Ej: >01:abcd67452301abff[CR]. Incluyendo la información del Manufacturer ID, el serial number, la versión y el tipo de dispositivo.
02	Application Address	Selecciona el meter desde el cual queremos recoger los datos. Ej: >02:abcd67452301[CR]. Siendo seleccionado el meter del comando anterior para el envio de datos. Los parámetros de entrada son iguales.
03	AES Key	Establece una contraseña de 16 bytes en hexadecimal. Ej: >03:09080706050403020l0a0b0c0d0e0f[CR]. Donde quedará establecida la contraseña tal que [9,8,7,6,5,4,3,2,1,10,11,12,13,14,15].
04	Access number	Establece el número de acceso, que combinado con el con la dirección del transmisor identifica el telegrama. Este parámetro se incrementa con cada transmisión asíncrona. Ej: >04:05[CR]. Donde se configura este parámetro con el valor 5.
05	Status Byte	Establece el bit de status del datagrama, el cual sirve como otro identificador del envio de datos. Ej: >05:0a[CR]. Donde se establece este byte con el valor 0a.
06	Contents of meter telegram	Este comando establece el valor del parámetro que define el contenido del telegrama entre los diferentes tipos definidos anteriormente en este documento. Ej: >06:08[CR]. Donde se marca el contenido de los

		datagramas a telgramas estáticos.
07	Transmit mode	Establece el modo de transmisión de datos entre síncrono y asíncrono. Ej: >07:00[CR]. Donde se ha marcado el modo ded transmisión como asíncrono.
08	Transmit power	Establece la potencia la potencia de transmisión entre una serie de valores, definidos anteriormente en este documento. Ej: >08:06[CR]. Donde se establece la potencia del envio de la señal, en el ejemplo mostrado anteriormente. Se marca la potencia a +10 dBm.
09	Module information	Nos da información sobre la versión HW y la versión FW utilizada. Ej: >09?[CR]. Donde una respuesta posible sería <1.151.06. Indicando primero la versión de la PCB usada, en este caso 1.15, y mas tarde la versión del FW siendo esta 1.06
0a	Sleep mode control	Determina el estado del dispositivo, entre Deep Sleep, despertado o reset al dispositivo. Usa diferentes códigos de configuración, los cuales se han expuesto anteriormente en este documento. Ej: >0a:00[CR]. //Pone el dispositivo a dormir. O >0a:02[CR]. //Aplica un reset software al dispositivo.
0b	Set MUC Flags	Establece las banderas de los nodos collector. Los bits del 7 al 4 están reservados pero no se usan. Los bits 3 y 2 establecen los niveles level0 y level1 respectivamente. El bit 1 configura si la transmisión de paquetes CNF-IR está activa o no. Y por último el bit 0 activa o desactiva la recepción ACK. Ej: >0b:0F. //Actival el level0, el level1, el CNF-IR y el ACK.
0c	Battery status	Da información sobre el nivel de la batería conectada a la placa. Ej: >0c?[CR]. // Nos devuelve un Battery OK si está dentro de los límites establecidos, y un Battery low si los niveles de batería están debajo de lo establecido.
0d	Serial number	Nos da información por debajo sobre el serial number del dispositivo elegido. Ej: >0d?[CR]. // Donde nos devuelve un número de 6 dígitos hexadecimales que nos da información del serial number. En este ejemplo podría ser <1234AB[CR].

Estos comandos, después de probados en el Wireless M-Bus software solution, nos permiten configurar los parámetros de los diferentes dispositivos. En este caso, un meter y un collector.

Tras las pruebas podemos concluir que hay comandos comunes para los dos tipos de nodos, como pueden ser el"0a" o el "00", que nos permiten poner a dormir o despertar a los dispositivos o nos dan la opción de establecer el modo de comunicación entre estos dispositivos. Y por el contrario hay ciertos comandos que están definidos para uno de los dos tipos de dispositivos. Como por ejemplo el comando 06 sólo establece el tipo del contenido de datagrama que enviarán los meter, por lo que podemos concluir el uso particular de estos dispositivos, y no tiene sentido usarlo para los collector. O el comando "0b", que activa o desactiva ciertas banderas particulares de los nodos collector.

Es importante mencionar también, que para la visualización de la información de forma correcta, hay diferentes opciones en función de como usemos el protocolo. Si estamos usando el protocolo con el Wireless M-Bus software solution, necesitamos habilitar una opción llamada "Sniffer", el cual nos permite ver la comunicación entre los dos tipos de nodos. Los comandos mencionados anteriormente también tiene opciones para la configuración de estos sniffers. No los hemos reflejado en los anteriores comandos, debido a que la opción que nos interesa a nosotros será la comentada a continuación.

La otra opción se basa en el software Stack de Wireless M-Bus que comentamos con anterioridad. Este Stack está diseñado para ser lanzado por Texas Instruments®. Y en este caso, aunque se usan los mismos comandos y la configuración es idéntica, no consta de los "sniffers" mencionados anteriormente, aunque que con un dispositivo UART TTL como el que se puede ver en la imagen a continuación, podemos visualizar la información enviada y recibida por el MCU de los dispositivos.

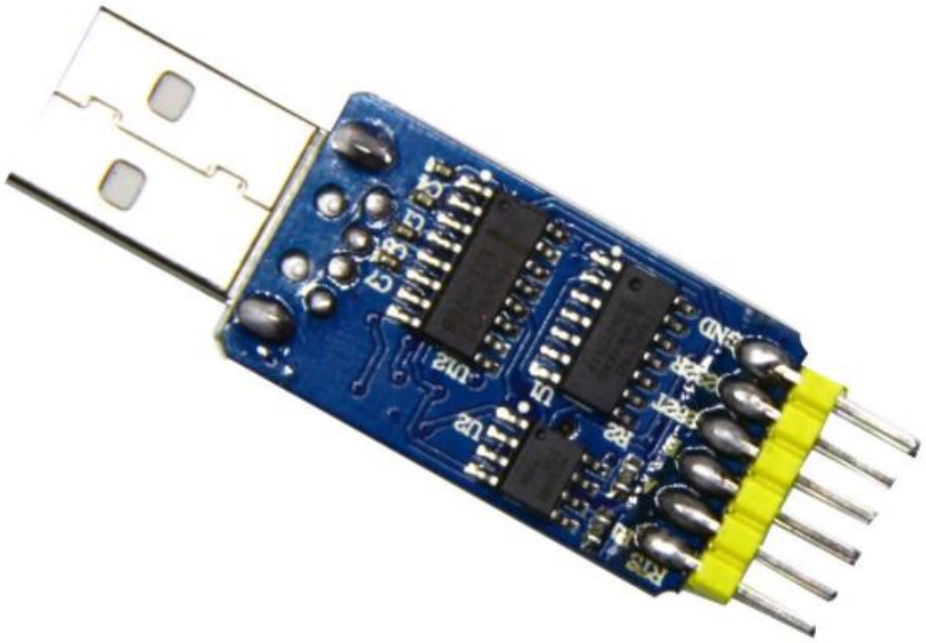


Figura 5-4. UART TTL para la visualización de datos.

Como conclusión, estos comandos junto al Wireless M-Bus Software Stack que incluye las librerías necesaria para la comunicación entre las diferentes capas del protocolo, y que provee al módulo con el firmware necesario para la correcta comprensión y procesado de los comandos mostrados nos permitirá programar una aplicación real, donde uno o más nodos meter se conecten a

un nodo collector, el cual se encargará de centralizar toda la información recibida y subirla a un servidor o una base de datos.

Para esta aplicación se usan dispositivos de medida y un servidor "virtual". Ya que no se dispone de dichos elementos, lo que se realiza en este proyecto hasta que se pueda realizar la prueba con todos los elementos, será la aplicación intermedia y la conexión mediante los comandos vistos anteriormente y el SW Stack de los elementos que actúan como meter y collector.

6.6 Descripción de los drivers y pruebas.

Se procederá ahora a explicar los drivers realizados para la correcta interpretación de los comandos y datos. Primero, comenzaremos hablando del driver utilizado para entender los comandos del protocolo Wireless M-Bus.

En este archivo, se plantean las funciones necesarias para la recepción de los datos, la descomposición de los datagramas, el añadido de nuevos dispositivos meter a la lista y su reserva en memoria.

- Primero se necesitará de una función que exija a collector a añadir el meter necesario a la lista, el cual reservará espacio en memoria para dicho dispositivo. Ya que Wireless M-Bus permite el conexionado de 32 nodos meter a un collector, por lo que tiene que reservar el espacio. Esto se hace con el comando visto anteriormente de identificador 01.
- Se requiere de una función adicional, para seleccionar de que meter concretamente estamos solicitando la información. Ya que si tenemos espacio reservado para diferenets nodos y decidimos pedir la información sobre uno de estos nodos el sistema no sabrá cual elegir, o incluso nos llega información de un nodo sin tener seleccionado con anterioridad, este no sabrá de donde le llega la información, ya que los identificadores de paquete de este protocolo no incluyen necesariamente el identificador de dispositivo. Esto se realiza con el comando de identificador 02.
- Ahora, ya suponiendo que tenemos registrado los diferentes meter y seleccionado el actual. Pediremos la información desde el nodo collector de dos formas diferentes. Si estamos usando un modo de envio de datos bidireccional, podremos exigir los datos desde el nodo collector al meter. Sin embargo, si es unidireccional tendremos que haber configurado el meter con un tiempo de envio de datos. En los cuales mientras tanto, permanecerán en modo Deep Sleep.
- Es importante mencionar, que el modo Deep Sleep en este driver está forzado cuando vence un timeout de 10 segundos después de la recepción de un paquete, ya que de la parte de descomposición de paquetes para su procesado se encarga el programa principal. Esto se realiza con el comando 0a:00[CR].
- También es importante mencionar la función de reset del dispositivo, el cual se realiza con el mismo comando que la anterior función, pero con un identificador diferente y puede ejecutarse en cualquier momento. 0a:02[CR].

Ahora vamos a hablar del driver que respecta la parte Nb-IoT. Esta es la parte después del procesado de la información por parte del micro, y la parte en la que envía la trama a internet.

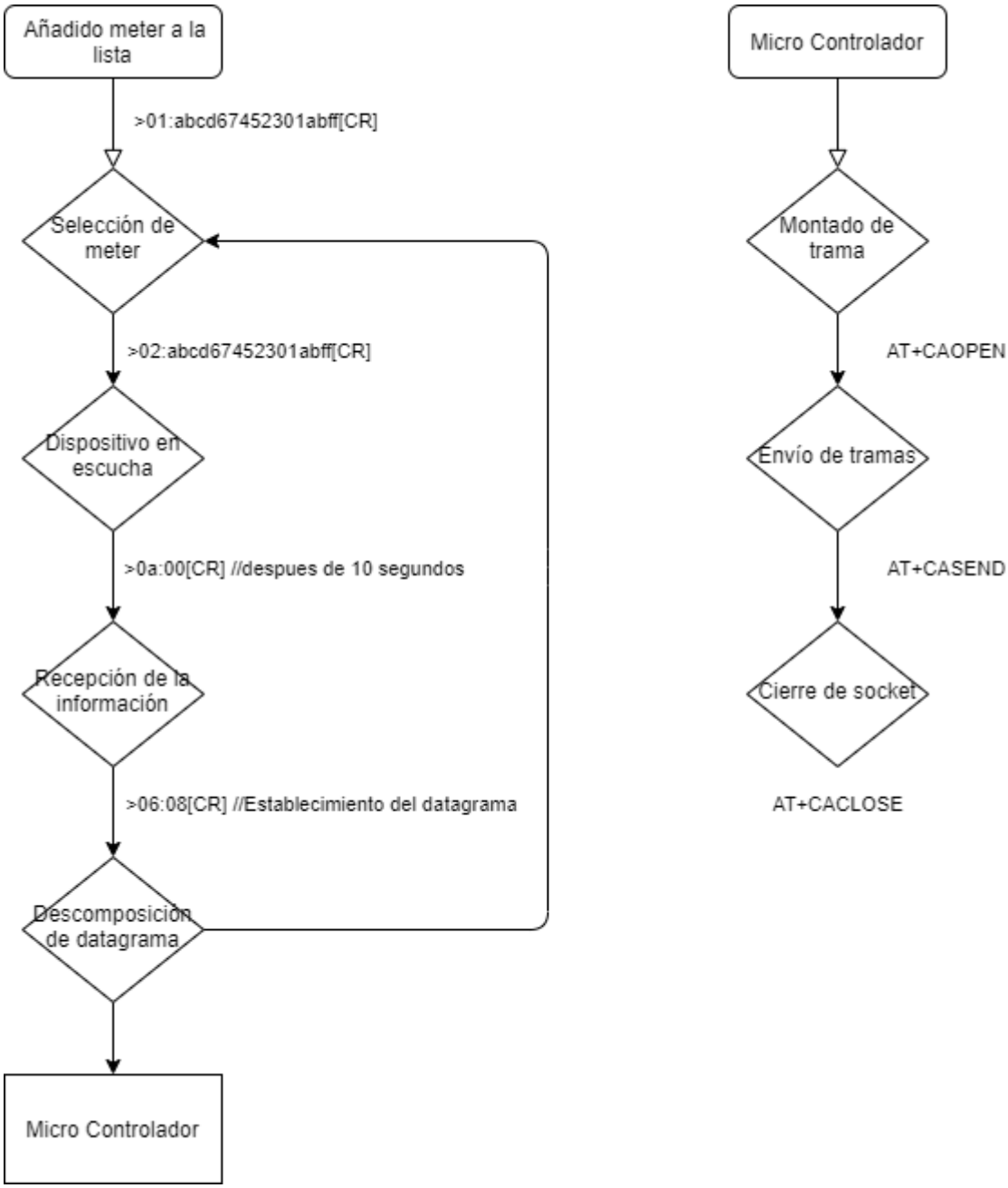
- Ya que toda la parte de configuración de los módulos viene de parte de Wireless M-Bus, en este driver no se incliyen funciones de dormido y despertado de los módulos, de reset del mismo o de configuración. En este archivo sólo se incluye la parte envio de tramas por Nb-IoT después de haber procesado la información recibida por Wireless M-Bus.
- Existe una función de inicio de conexión TCP/UDP, en la que especificando el servidor y puerto donde queremos enviar la información, abre un socket del tipo que nosotros definamos. En esta misma función se produce el montaje de la trama. Esta función se realiza con el comando AT+CAOPEN, donde se especificarán a continuación todos los parámetros.
- Otra función vital es la de envio de la información a través del socket abierto con la función anterior. En esta función sólo se especificará la trama a enviar, montada también con anterioridad y el número del socket, ya que podemos tener varios abiertos simultáneamente. Esta se realiza con el comando AT+CASEND.
- Otra función necesaria será la de comprobar el estado de conexión del socket, para ver si está funcionando debidamente y si somos capaces de mandar la conexión al servidor. Esta función se realiza con el comando AT+CASTATE. Nos devolverá un 1 si el servidor está abierto y esperando para la recepción de los datos.
- La ultima función importante en este driver consistirá en cerrar el socket si esta funcionando mal o ya no requerimos de su uso. Esta función tendrá como parámetros de entrada el identificiador del socket establecido en el inicio de la función y se realiza con el comando AT+CACLOSE.

Se han realizado unas pequeñas pruebas de funcionamiento relacionadas con el siguiente diagrama de flujo, el cual utiliza los driver de entrada de datos por Wireless M-Bus y envio al servidor por Nb-IoT.

Corresponde el diagrama de la derecha al procedimiento de recepción de información por Wireless M-Bus, el examinado de las tramas para su extracción de información. Se observa como las tareas realizadas corresponden con la parte del código que está reflejada en el main.c, encargado de las tareas principales, y las ordenes intermedias serían las realizadas por el driver Wireless M-Bus.

Por otro lado, en la imagen de la derecha se puede ver un secillo diagrama de flujo, el cual muestra como la información recibida desde la UART, es procesada a través de otros comandos AT por el moderm Nb-IoT para su envio al servidor.

Tras esta prueba, se ha observado la correcta transmisión de la información a través de estos dos protocolos, aprovechando la ventaja de ambos. Por una parte, se ha utilizado la capacidad de Wireless M-Bus de obtener información de varios dispositivos meter abaratando costes respecto a otros protocolos y facilitando la concentración de toda esta información en un mismo nodo. Y por el otro lado, gracias a Nb-IoT, con una sola suscrpción al operador, somos capaces de subir a internet toda esta información concentrada en un servidor.



Como conclusión, es necesario mencionar la viabilidad de realizar un sistema de tele-lectura de contadores de agua a través de un hub Wireless M-Bus y Nb-IoT, el cual ofrece una concentración de información alta, la posibilidad de operar en diferentes bandas, abaratando costes y dando también la posibilidad de subir la información a la red después de un procesado, el cual nos permitirá una gestión mucho más eficiente de los datos.

REFERENCIAS

[1] SALHER, Reutilizar el agua, una inversión para un futuro sostenible, A review. En: *https://www.retema.es/noticia/reutilizar-agua-una-inversion-para-un-futuro-sostenible-yhdYZ*

[2] Boletín Oficial del Estado, Plan nacional integrado de energía y clima 2021-2030, *https://www.boe.es/diario_boe/txt.php?id=BOE-A-2021-421*

[3] FAO, Estimaciones del caudal del agua, Disponible en: http://www.fao.org/tempref/FI/CDrom/FAO_Training/FAO_Training/General/x6705s/x6705s03.htm

[4] Gobierno de España, Contadores, Disponible en: https://www.mapa.gob.es/es/ministerio/servicios/informacion/plataforma-de-conocimiento-para-el-medio-rural-y-pesquero/observatorio-de-tecnologias-probadas/material-de-riego/contadores.aspx

[5] Communication systems for meters, EN 13757, Disponible en: https://ec.europa.eu/eip/ageing/standards/ict-and-communication/data/en-13757_en.html

[6] Norma Española UNE 82326, Protocolo de comunicación para lectura de dispositivos de contadores de agua y otros dispositivos de medida o control de instalaciones de agua, Disponible parcialmente en: https://www.en.aenor.com/normas-y-libros/buscador-de-normas/une?c=N0045451

[7] Texas Instruments CC1310 SimpleLink Ultra-Low Power Wireless MCU, Disponible en:

https://www.mouser.es/new/texas-instruments/ti-cc1310-simplelink-mcu/?gclid=Cj0KCQjwpdqDBhCSARIsAEUJ0hPLQaL75rG_ST7rZJPbch08QC28b0028KTcMGidIVkKxVCqI7tk5IIaApEsEALw_wcB

[8] LAUNCHXL-CC131O, Disponible en: https://www.ti.com/tool/LAUNCHXL-CC1310?utm_source=google&utm_medium=cpc&utm_campaign=epd-null-null-GPN_EN_EVM-cpc-evm-google-wwe&utm_content=Tool&ds_k=LAUNCHXL-CC1310&DCM=yes&gclid=Cj0KCQjwpdqDBhCSARIsAEUJ0hMJaxb4cp4jk0qGEsjr4GS82oSe0BKdy3EyGqwlnEXM930qMcra3FMaArGUEALw_wcB&gclsrc=aw.ds

[9] List of Wireless M-Bus Stack APL Interface, Disponible en:

file:///C:/ti/wmbus_cc13x0_rtos_2_0_0/Documentation/APL-Interface-Doc-v4.0.1/APL-Interface-Doc-v4.0.1/_a_p_l__i_n_t_e_r_f_a_c_e__l_i_s_t.html

[9] Wireless M-Bus Commands, Disponible en: https://static.iqrf.org/User_guide_wM-Bus_7xD_151209.pdf

GLOSARIO

IoT *Internet of Things*
SPI *Serial Peripheral Interface*
I2C *Inter-Integrated Circuit*
SoC *System on a Chip*
SNR Relación señal-ruido
RGB *Red, green, blue*
UVC *USB Video Class*
USB *Universal Serial Bus*
GPIO *General Purpose Input-Output*
HDMI *High-Definition Multimeda Interface*
CSI *Camera Serial Interface*
SSH *Secure Shell*
IP *Internet Protocol*
DHCP *Dynamic Host Configuration Protocol*
SSID *Service Set Identifier*
SRAM *Static Random Access Memory*
UART *Universal Asynchronous Receiver-Transmitter*
LiPo Litio y polímero
CMOS *Complementary Metal-Oxide Semiconductor*
VGA *Video Graphics Array*
QVGA *Quarter Video Graphics Array*
CIF *Common Intermediate Format*
QCIF *Quarter Common Intermediate Format*
LVDS *Low-Voltage Differential Signaling*
SMBus *System Management Bus*
PEC *Packet Error Checking*
SMIF *System Management Interface Forum*
SDA *Serial Data*
SCL *Serial Clock*
ACK/NACK *Acknowledgement / No Acknowledgement*
PDF *Portable Document Format*
UNE Una Norma Española