Universidad Zaragoza
1542

Trabajo Fin de Máster

Evaluación y caracterización experimental del consumo energético del dispositivo Waspmote para redes sensoriales y de comunicaciones

Lorena I. Florentín Martín

Director
Arturo Mediano Heredia
Departamento
Ingeniería Electrónica y Comunicaciones

Universidad de Zaragoza / Escuela de Ingeniería y Arquitectura
2014

Agradecimientos

Quiero agradecer este proyecto a toda mi familia, en especial a mi madre Josefina y mi padre Leonardo por todo el apoyo e ímpetu que me han dado desde que comencé el TFM.

También quiero agradecer a mi director D. Arturo Mediano Heredia por la comprensión que ha tenido conmigo durante todo este tiempo y por la ayuda que me ha prestado ya que es un excelente profesor y director de proyectos y trabajos fin de máster. Gracias por dedicarme tiempo y paciencia.

No me olvido de mi director en la entidad, Javier Solobera Abad por su implicación que ha tenido en todo momento y por la gran ayuda que he recibo de él.
Agradezco la implicación en el trabajo al resto de compañeros de Libelium (en particular a Manuel Calahorra, Yuri Carmona, Marcos Yarza y Alejandro Gallego)

Resumen

El propósito de este Trabajo Fin de Máster, realizado en la empresa Libelium Comunicaciones Distribuidas S.L., es la caracterización del consumo eléctrico de los distintos modos y opciones de funcionamiento de la plataforma Waspmote, incluyendo sus sensores y módulos de comunicaciones asociados, a través de medidas experimentales para poder realizar estimaciones de autonomía de su batería de alimentación realizando lo que se conoce como "Energy Guide" cada vez de mayor relevancia en productos electrónicos de todo tipo y, especialmente, en redes sensoriales inalámbricas.

El desarrollo del proyecto comprende, el estudio y la formación de la plataforma Waspmote, de las placas de sensores que la integran y de los módulos de comunicaciones, la familiarización del software de desarrollo, la caracterización de las medidas realizadas junto con el cálculo del consumo de cada una de ellas analizando los resultados obtenidos, el cálculo del consumo en procesos globales mostrándose la vida de la batería y finalmente, la redacción de la Guía Energética de usuario mostrándose los consumos de cada prueba.

Además, este trabajo nos permite integrarse en un entorno laboral en el ámbito empresarial, obteniendo una gran experiencia en el campo de las comunicaciones inalámbricas y del consumo energético que requiere cada una de ellas.

Este TFM servirá a la empresa Libelium como base para futuros proyectos basados en el consumo energético de redes sensoriales inalámbricas.

ÍNDICE

1. Introducción

1.1. Marco de trabajo

Este Trabajo Fin de Máster (TFM), aborda la caracterización del consumo energético de todas las funcionalidades del dispositivo *Waspmote* de la empresa **Libelium Comunicaciones Distribuidas S.L.** y de toda su gama de productos sensoriales y de comunicaciones.

Se ha realizado bajo la Dirección del Dr. Arturo Mediano Heredia, Profesor Titular del Departamento de Electrónica y Comunicaciones de la Universidad de Zaragoza, entre Septiembre del 2013 y diciembre del 2014 contando con el apoyo y supervisión de D. Javier Solobera Abad, Subdirector del Departamento de I+D de Libelium.

El trabajo se ha llevado a cabo, íntegramente, en las instalaciones de Libelium en el Centro Europeo de Empresas e Innovación de Aragón en Zaragoza acogido a un convenio de colaboración entre la Universidad de Zaragoza y la empresa gestionado por el servicio UNIVERSA.

Libelium nació en 2006 como empresa spin-off de la Universidad de Zaragoza. La empresa dedica al diseño y fabricación de hardware específico para la implementación de redes sensoriales inalámbricas, redes malladas y protocolos de comunicación para todo tipo de redes inalámbricas distribuidas. Esta empresa exporta ya su tecnología a 75 países y ha conseguido multiplicar por doce las ventas en los últimos cuatro años. Sus esfuerzos se han visto reconocidos con diversos premios.

1.2. Redes de Sensores inalámbricas y la importancia de medir su consumo.

Las redes sensoriales inalámbricas [1], [2], [3], más conocidas como WSN, son redes enfocadas a la monitorización del entorno.

Están compuestas de pequeñas máquinas, equipadas con diversos sensores, que trabajan de forma colaborativa a la hora de enviar los datos captados. Estas maquinas son denominadas *'motes'* o nodos. Se le llaman así por su ligereza y reducido tamaño.

Los motes son dispositivos electrónicos capaces de captar información proveniente del entorno en el que se encuentran, procesarla y transmitirla inalámbricamente hacia otro destinatario.

El origen de este tipo de redes es militar. Tener conocimiento en tiempo real del campo de batalla es esencial para el control, las comunicaciones y la toma de decisiones. Ya desde la Guerra Fría se usan para detectar submarinos [2], pero últimamente las áreas en las que se aplican son muy diversas:

- **Medio ambiente.**

  Los sensores se emplean para el medio ambiente en el caso de incendios forestales, detección de inundaciones y exploración de animales en su hábitat natural.

  Se controla desde la temperatura, humedad relativa (HR) y nivel de contaminación en campo y ciudad, hasta prevenir y avisar de posibles desastres como incendios o actividad sísmica. Se monitorizan zonas muy extensas de forma remota, instantánea y sin el impacto de la presencia humana.

- **Salud :**

  En el ámbito de la salud, las redes de sensores pueden llevar a cabo acciones que controlen pacientes, diagnostiquen enfermedades, administren la medicina, detecten y controlen el movimiento de pacientes dentro del hospital y demás funciones.

- **Agricultura y ganadería:**

  Para gestionar los cultivos es fundamental conocer las variables ambientales en cada momento, puesto que esto ayuda a una mejor toma de decisiones. En explotaciones intensivas de tipo invernadero, las redes distribuidas ayudan automatizar los procesos. Al aplicar el riego exacto en el momento justo, el ahorro de agua y otros recursos es considerable y la producción óptima, tanto en cantidad como en calidad. Combinando varios parámetros se prevén posibles enfermedades o plagas, y observando detenidamente la causa- efecto se llega a una mejor comprensión de la naturaleza.

- **Industria:**

  Sistemas de control de calidad de producción y automatización.

- **Entornos de alta seguridad:**

  En aeropuertos o centrales nucleares, por ejemplo, como complemento a otros sistemas de seguridad (cámaras de vigilancia, etc).

- **Tráfico:**

  Los nodos se complementan con las cámaras para prevenir atascos y evitar o avisar de accidentes.

- **Domótica y ahorro energético:**

  Cada vez se implantan más motes en las viviendas para hacer la vida más cómoda y además ahorrar energía gestionando de forma inteligente la calefacción/aire acondicionado o iluminación.

Enumeramos a continuación algunas de las ventajas generales de ese tipo de redes de sensores inalámbricas [2] y [4]:

• **Sin cables**. El hecho de que la comunicación sea inalámbrica y la alimentación se apoye en baterías y energía solar facilita una implantación rápida y barata, ahorrando en cableado.

• **Auto-organización**. Los protocolos sobre los que trabajan los motes actualizan sus tablas de enrutamiento periódicamente, sin ser necesario acceder manualmente. Esto hace la red muy flexible, pudiendo cambiar la ubicación de los motes.

• **Escalabilidad**. Los nuevos nodos son reconocidos automáticamente por la red.

• **Robustez**. Si uno a varios nodos no están disponibles, la gran variedad de caminos para realizar las transmisiones hace posible que la red siga funcionando. Es la ventaja de ser un sistema dinámico o no centralizado.

• **Fiabilidad**. Existen mecanismos que aseguran la buena recepción de cada paquete.

• **Seguridad**. Se cuenta con algoritmos de cifrado para proteger las comunicaciones.

• **Mínimo mantenimiento**. Con su bajo consumo, las baterías y paneles solares de los motes les permiten funcionar indefinidamente.

Como se ha comentado anteriormente, los nodos o 'motes' tienen, normalmente, como fuente de alimentación baterías o paneles solares. Ante la limitación de la vida útil del dispositivo hay que realizar una gestión eficiente del consumo energético. El consumo de energía viene dado por lo que consumen las funcionalidades del mote, los sensores, y la comunicación. La mayor cantidad de energía es consumida en la transmisión de información, siendo menor en el procesado.

La eficiencia energética [5] y [6] es crucial en redes de sensores inalámbricas. Su objetivo es maximizar el tiempo de vida de la red al mismo tiempo que la aplicación cumple con sus requisitos de *QoS* (calidad de servicio). Las mejores tecnologías que permiten aumentar la capacidad de las baterías progresan despacio. Esto quiere decir que la eficiencia energética seguirá siendo un reto para este tipo de redes en el futuro próximo.

Diseñar los nodos para un bajo consumo supone elegir componentes específicos. El primer parámetro a considerar es el consumo de energía de la CPU, el sensor, el transceptor y, posiblemente, de otros elementos, como la memoria externa y los periféricos durante el modo normal de operación.

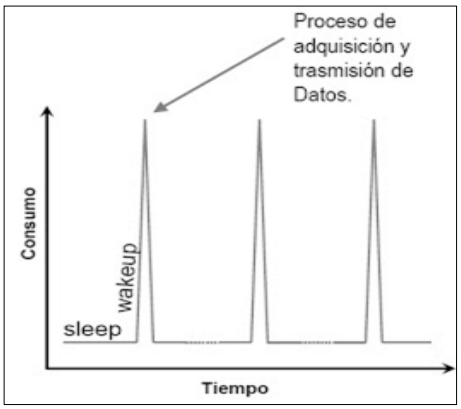


Fig. 1.1. Esquema simplificado del consumo de un nodo en sus tres estados básicos de trabajo. [3], [6] y [7]

La elección de elementos de baja potencia implica normalmente aceptar compromisos sobre el rendimiento medio. Por regla general, una CPU de baja potencia opera en un ciclo reducido de reloj, con menos características en el chip que otras unidades homólogas que consumen más energía.

La optimización del consumo de energía en los nodos para lograr el máximo tiempo de vida de la red, es un objetivo básico. Los elementos a considerar son:

- Economizar la distancia de las comunicaciones.
- Técnicas de software: programación eficiente de líneas de código.
- Protocolos de enrutamiento.
- Estrategias hardware de ahorro de energía.

Para el ahorro de energía, los nodos pasan habitualmente por estos estados (Fig. 1.1):

- *Sleep:* El nodo pasa la mayor parte del tiempo en este estado sin actividad.
- *Wake-up:* Debemos de minimizar este tiempo de arranque para pasar rápidamente al estado de trabajo.
- *Active:* Idealmente de la menor duración posible realizando la tarea especificada pero retornando de inmediato al estado *sleep*.

El bloque de comunicación inalámbrica es el primer contribuyente al consumo de energía. Un sistema distribuido significará que algunos sensores necesitarán comunicarse a través de largas distancias, lo que se traducirá en mayor consumo. Por ello, es una buena idea el procesar localmente la mayor cantidad de energía, para minimizar el número de bits transmitidos.

La CPU es capaz de quedar en estado "*sleep*" mientras "no tenga nada que hacer".

El envío de datos desde los nodos puede ser de tres formas: de **modo continuo** en los intervalos establecidos, **dirigido por eventos** (envía cuando se cumple cierta condición), o dirigido **por consulta** (sólo cuando le solicita). También hay sistemas híbridos que utilizan una combinación de los antes mencionados.

Ese perfil de consumo es crítico para poder disponer de sistemas eficientes y, su evaluación en una gama concreta de sensores inalámbricos es el objeto de este TFM.

En ese campo de aplicaciones, la empresa aragonesa Libelium hizo pública una nueva plataforma de dispositivos, llamada Waspmote, para la creación de Redes Sensoriales Inalámbricas. Estas redes permiten controlar por medio de cientos de sensores interconectados inalámbricamente

lo que está sucediendo en tiempo real en un entorno determinado. Entre otras cosas permiten detectar incendios forestales cuando son colocados en las copas de los árboles, o incluso las crecidas repentinas del nivel del agua cuando se despliegan por las orillas de los ríos.

En entornos urbanos encuentran varios campos de actuación. Por un lado, en la detección de calidad del ambiente, puesto que permiten medir la cantidad de agentes contaminantes tales como CO, CO2. Por otro lado, posibilitan la creación de redes de control energético "*smart metering*" que permiten conocer la energía que se está utilizando en un punto determinado y disminuir así consumos no necesarios.

Estos dispositivos sensoriales se comunican entre sí mediante sistemas inalámbricos de distintas frecuencias, lo que les permite evitar las interferencias que causan las redes Wifi en entornos urbanos.

Para disminuir el consumo de las comunicaciones se han creado protocolos basados en los mecanismos de comunicación usados por enjambres de insectos tales como abejas y luciérnagas. El concepto es que toda la red esté sincronizada de forma que todos los dispositivos sensoriales sean capaces de enviarse la información al mismo tiempo.

Waspmote fue, el resultado de 2 años de investigación por parte de Libelium. El mayor reto de esta plataforma fue disminuir el consumo de los dispositivos al mínimo, mientras que se aumentaban sus posibilidades de comunicación y de integración de nuevos sensores, ya que una de las principales novedades está en la eficiencia energética, que es crucial en redes de sensores inalámbricos, como se ha comentado anteriormente.

Un consumo típico de tan solo 0.07µA en estado de reposo permite, a los nodos sensoriales, vivir durante años ininterrumpidamente y que la fuente de alimentación, la batería, permite tener mayor tiempo de aguante sin ser sustituida por otra. Por otro lado, la integración módulos de comunicación de alta sensibilidad permite una comunicación de varios kilómetros entre los dispositivos.

Ante esa problemática, es fundamental para el desarrollador/cliente disponer de una Guía Energética, que le permita minimizar el consumo de su sistema en una aplicación final escogiendo el sensor más adecuado, calculando su vida útil, los costes de mantenimiento (desplazamientos para reemplazar la batería), etc. .

Quizás, lo más importante de todo es que, con esa guía, los desarrolladores tomarán conciencia de qué tipo de código o estrategia de trabajo es eficiente y cuál no. Por ejemplo, no tiene sentido en muchas aplicaciones medir la temperatura y enviar el dato usando el estándar ZigBee muy a menudo (e.g. cada 10 segundos). Esta es una práctica habitual de muchos usuarios finales, sorprendiéndose de que la batería les proporciona no más de cinco días de autonomía.

1.3. Dispositivo Waspmote.

Este proyecto trabaja con el dispositivo Waspmote PRO v1.2, fabricado y distribuido por Libelium Comunicaciones Distribuidas S.L.

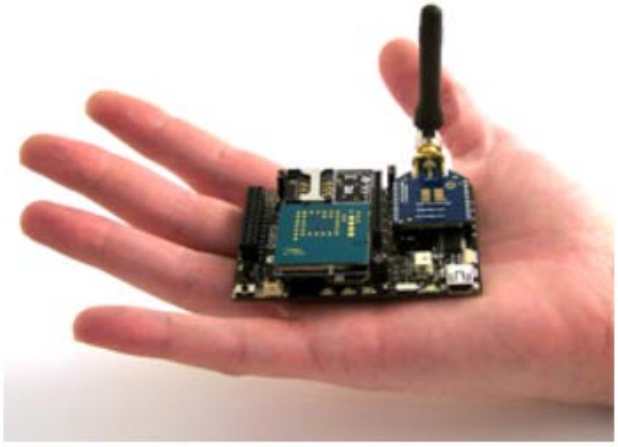


Fig. 1.2. Dispositivo Waspmote [8]

Waspmote es un dispositivo sensorial especialmente orientado a desarrolladores. Puede trabajar con diferentes protocolos (ZigBee, Bluetooth, GPRS) y frecuencias (2.4GHz, 868MHz, 900MHz) siendo capaz de establecer enlaces de hasta 12km.

Cuenta con un modo de hibernación con un consumo de 0.07 µA que permite ahorrar batería cuando no está transmitiendo.

El dispositivo *Waspmote* dispone también de más de 50 sensores como temperatura, humedad o radiación entre otros, y de un entorno de desarrollo (IDE) completamente *open source* (IDE + librerías + compilador).

El propósito de este dispositivo es facilitar la configuración de redes sensoriales en multitud de escenarios diferentes, desde una red de contadores industriales o una red de sensores en cultivos, hasta una red de sensores en un lugar público.

A continuación se observa dos esquemas de la organización hardware y software. En la figura de la izquierda, se observa cómo diferentes módulos se acoplan al dispositivo *Waspmote* a través de distintas conexiones. La figura de la derecha, muestra cómo están estructuradas las bibliotecas en el entorno de desarrollo.

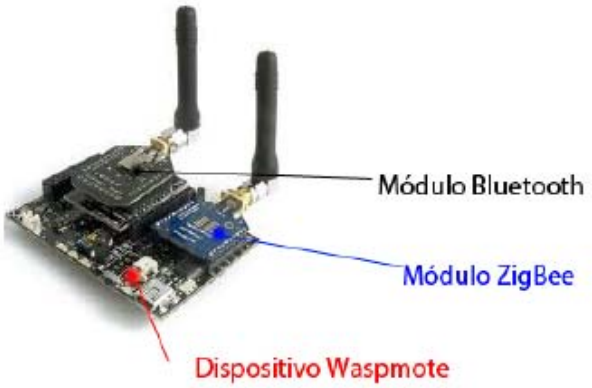


Fig. 1.3. Organización hardware

Fig. 1.4. Organización software

En el anexo A se explica con más detalle el diseño del dispositivo Waspmote.

1.4. Motivación del proyecto.

Desde un punto de vista general, la motivación de este proyecto surge de la necesidad de una empresa aragonesa, Libelium, de desarrollar una guía técnica para ver el consumo energético que se produce al ejecutar cada acción que compone cada ejemplo medido tanto de las funcionalidades del Waspmote con cada uno de sus, aproximadamente 60 sensores y 10 módulos de comunicaciones, ya que el mote funciona con baterías y éstas tienen una carga limitada.

Existen numerosas funcionalidades del Waspmote (leds, EEPROM, RTC, hibernación, estado de la batería, tarjeta SD, acelerómetro, etc.), sensores (temperatura, humedad, estación meteorológica, presión atmosférica, dendrómetros, iluminación, etc..) y módulos de comunicaciones (ZigBee, Bluetooth, Xbees, GPS, GPRS, etc..). El consumo energético correspondiente a cada uno, su caracterización y el impacto que se produce en la batería usar una determinada combinación de opciones es un reto muy interesante.

Desde un punto de vista personal, tanto la necesidad real de una empresa como la actualidad de esta temática (sensores inalámbricos y eficiencia energética) me motivaron a la hora de realizar este trabajo.

1.5. Objetivos del proyecto.

El objetivo principal de este trabajo es la caracterización del consumo eléctrico de los distintos modos y opciones de funcionamiento de la plataforma Waspmote, incluyendo sus sensores asociados, a través de medidas experimentales para poder realizar estimaciones de autonomía de su batería de alimentación realizando lo que se conoce como "*Energy Guide*" (ver Anexo F y el documento está en el CD adjunto) cada vez de mayor relevancia en productos electrónicos de todo tipo y, especialmente, en redes sensoriales inalámbricas.

Para esto se propuso cumplir, en orden, los siguientes objetivos parciales:

- **Realizar un estudio inicial**: búsqueda de información, leer documentación de dispositivos, guías técnicas del HW a utilizar (Waspmote, placas de sensores, módulos de comunicaciones), informarme sobre las redes de sensores inalámbricas (WSN), etc.
- Familiarización con el módulo IDE (compilador).
- Familiarización con el software: estructura habitual.
- **Definir qué variables y de qué forma** se van a medir.
- **Elección de la instrumentación** necesaria para realizar las mediciones.
- **Definición de la estrategia de medida del consumo** de Waspmote.
- Cargar los códigos en Waspmote y medir su consumo. Esta es la tarea principal. Aquí se puede modificar lo necesario para evitar consumos irreales, añadir líneas de código para correcciones de posibles *bugs* que vayan saliendo o para el cálculo del consumo con mayor precisión.
- **Optimización del código** cuando resulte necesario.
- Calcular el tiempo total de cada bloque de código y estudiar la altura media total de

consumo.

- Finalmente, **elaboración de la documentación** técnica que incluye redacción de la memoria y la guía técnica energética de usuario.

Aparte de exponer el objetivo principal que se busca y los parciales, se le quiere hacer saber al cliente, el impacto que tiene, en la batería de Waspmote, usar ese determinado producto, es decir, según el consumo que se produzca en ese determinado producto, si el código utilizado será o no eficiente, y así mostrarle si la batería va a tener más tiempo de aguante para no ser sustituida por otra en corto tiempo y evitar que el cliente, por ejemplo, mida más veces un producto que requiera de mucho consumo, porque mucha gente lo hace, sin darse cuenta del gran impacto que tendrá en su batería y luego se sorprenden de que les aguanta muy poco. Así, servirá para calcular su vida útil y costes de mantenimiento de la misma.

1.6. Desarrollo del proyecto

El trabajo se ha desarrollado en diferentes etapas, las cuales se presentan a continuación:

- **Generación de especificaciones** de acuerdo a las necesidades de la empresa.
- **Estudio del hardware Waspmote** (incluyendo su gama de sensores y módulos).
- **Análisis del problema** y **revisión bibliográfica**.
- **Revisión de especificaciones**.
- **Familiarización con el software** de desarrollo.
- **Selección y puesta en marcha de la instrumentación** requerida.
- Desarrollo del trabajo de **medida y análisis de resultados**.
- Redacción del documento “**Energy Guide**”.
- Redacción de la **Memoria de TFM**.

2. Medida de consumo en redes de sensores inalámbricas.

2.1. Estado del Arte

Durante la última década han aparecido infinidad de dispositivos que utilizan sensores para el monitoreo y control [9]. Gracias a la evolución en prestaciones y tamaño que nos ofrece la electrónica actual, disponemos de sensores capaces de comunicarse de manera inalámbrica, capacidad de procesamiento y autonomía propia. Este tipo de dispositivos nos abre un nuevo abanico de oportunidades para diseñar y crear todo tipo de aplicaciones, protocolos y sistemas capaces de facilitar el trabajo a los seres humanos a la vez que reducen sus costes.

Las Redes Inalámbricas de Sensores o RIS (también llamadas WSN por sus siglas en ingles) pueden estar compuestos por decenas, cientos o incluso miles de pequeños computadores que operan con baterías, llamados motes y que son distribuidos a lo largo de un ambiente de interés particular. Cada nodo en una red ad-hoc recolecta datos de su ambiente, como la cantidad de luz, temperatura, humedad, vibraciones y otros factores ambientales.

Las WSNs constituyen una tecnología emergente de adquisición de datos que recientemente está creando un gran interés gracias a sus posibilidades, siendo aplicadas en numerosos ámbitos científicos e industriales para la realización de estudios y control de procesos. El uso de comunicaciones inalámbricas permite que los dispositivos sensores que la forman sean emplazados fácilmente sobre el terreno. Esto, unido al bajo coste de algunos dispositivos, permite el despliegue de un gran número de dispositivos que actúan como nodos de la red.

Que las WSN sean una tecnología emergente es debido, en gran medida, a los recientes avances en los sistemas micro-electro-mecánicos (MEMS, Micro-Electro-Mechanical Systems), las comunicaciones inalámbricas y la electrónica digital. Estos avances han permitido el desarrollo de nodos sensores multifuncionales de bajo coste, baja potencia, pequeño tamaño y que permiten comunicarse inalámbricamente entre alcances cada vez mayores.

Este tipo de redes permiten monitorizar entornos difícilmente accesibles, trabajando de forma coordinada para realizar el control de los parámetros deseados.

Dentro de las investigaciones realizadas sobre las redes inalámbricas sensoriales o WSN podemos encontrar estudios que se centran en aspectos como la eficiencia energética que permiten aumentar el tiempo de vida útil de los dispositivos.

El objetivo de la eficiencia energética [6] es maximizar el tiempo de vida de la red al mismo tiempo que la aplicación cumple con sus requisitos de ¨QoS. Las mejoras tecnológicas que permiten aumentar la capacidad de las baterías progresan despacio. Esto quiere decir que la eficiencia energética seguirá siendo un reto para este tipo de redes en el futuro próximo.

Diseñar los nodos para un bajo consumo, supone elegir componentes de baja potencia. El primer parámetro a considerar es los consumos de energía de la CPU, el sensor, el radiotransceptor y posiblemente, de otros elementos, como la memoria externa y los periféricos durante el modo normal de operación.

La elección de elementos de baja potencia implica normalmente aceptar compromisos sobre el rendimiento medio. Por regla general, una CPU de baja potencia opera en un ciclo reducido de trabajo, con menos características en el chip que otras unidades homólogas que consumen más energía.

La optimización del consumo de energía en los nodos [5] para lograr el máximo tiempo de vida de la red, es un objetivo básico.

Los elementos a considerar son: la comunicación es el primer consumidor de energía. Un sistema distribuido significará que algunos sensores necesitarán comunicarse a través de largas distancias, lo que se traducirá en mayor consumo. Por ellos, es una buena idea el procesar localmente la mayor cantidad de energía, para minimizar el número de bits transmitidos. La CPU es capaz de quedar en estado “sleep” mientras “no tenga nada que hacer”. El envío de datos desde los nodos puede ser de tres formas: de modo continuo en los intervalos establecidos, dirigido por eventos (envía cuando ose cumple cierta condición) o dirigido por consulta (solo cuando se le solicita).

También hay sistemas híbridos que utilizan una combinación de los antes mencionados.

- Economizar la distancia de las comunicaciones.
- Técnicas de software: programación eficiente de líneas de código
- Protocolos de enrutamiento.
- Estrategias hardware de ahorro de energía.

Como se ha dicho anteriormente, el consumo energético ha sido un tema de estudio durante mucho tiempo en estas redes de sensores inalámbricas. Al pasar el nodo por diferentes estados, se han ido estudiando las diferentes maneras de cómo tener el nodo para reducir su consumo. Se considera minimizar el tiempo *wake up* del mismo para pasar rápidamente al estado de trabajo y se intenta minimizar el tiempo de activo del nodo para retornar de inmediato al estado *Sleep*. Mantener el nodo en este estado es crucial para evitar pérdidas energéticas. Por eso, últimamente se diseñan nodos de sensores con un consumo extremadamente bajo (ordenes de uA) en estado de reposo, que permiten a los nodos sensoriales vivir durante años ininterrumpidamente y que la fuente de alimentación, la batería, permita tener mayor tiempo de aguante sin ser sustituida por otra.

Una de las grandes limitaciones de las redes de sensores, es la disponibilidad de esa energía, por ello se han ido estableciendo estrategias para gestionar adecuadamente el consumo energético disponible y maximizar la consecución de más energía.

Las redes de sensores típicamente almacenan su energía en baterías, las cuales tiene una capacidad finita. A veces es muy dificultoso reemplazar las baterías si se agotan, por lo que en aplicaciones pensadas para funcionar de forma ininterrumpida durante un largo periodo de tiempo, es necesario que los nodos consigan por ellos mismos energía con la que autoabastecerse. Esta necesidad ha dado lugar a una línea de investigación llamada “Energy harvesting” o “Energy Scavenging”. Una de las fuentes de energía más utilizadas es la solar. Esto se debe a que tanto la fuente de energía como la tecnología están muy extendidas y se obtienen rendimientos muy altos comparados con otras fuentes. Se han llegado a proponer paneles solares aplicados a nodos de redes de sensores para obtener la máxima energía con la que recargar una batería. Para ello, emplean un convertidor DC/DC variable controlado por un microcontrolador. Además, también ha habido intentos de extraer energía de las señales de RF en el ambiente o idear un sistema analógico para aprovechar la energía de emisiones de radio y cargar un condensador con el que poder alimentar un pequeño nodo.

Ha habido muchas más propuestas de fuentes de energía ciertamente imaginativas.

Se suelen utilizar sistemas operativos como el software base sobre el cual desarrollar una aplicación de un nodo. Los motivos son: la velocidad de desarrollo y la reutilización de código de terceros.

Actualmente, los sistemas operativos para redes de sensores más utilizados son TinyOS, Contiki y Mantis OS. El objetivo principal de estos sistemas operativos es proporcionar un mecanismo robusto y confiable de operación, manteniendo a la vez el menor consumo posible en

cada parte del nodo y en cada momento.

Al reducir la potencia de cada parte del nodo, se reduce el consumo global y, por tanto, se extiende el tiempo de vida de la batería de los motes.

Al estudiarse varios sistemas operativos que ayudan a conseguir un menor consumo energético, TinyOs es el más asequible para ello. Sigue el modelo de programación dirigida por eventos, consigue que el manejo de la energía dl nodo sea muy eficiente.

Resumiendo, el consumo energético en redes de sensores inalámbricas ha estado presente desde que se diseñaron este tipo de redes. El estudio de diferentes formas y aplicaciones para reducir su consumo, han aumentado cada vez más hasta la actualidad y aun se siguen investigando. Se dice que se seguirá un camino infinito. La eficiencia energética de la que se ha estado hablando, es uno de los asuntes más importantes que se han tenido en cuenta en las redes de sensores porque cuando más se consiga bajar el consumo de un nodo, mayor será el tiempo durante el cual puede operar y mayor tiempo de vida tendrá la red.

La transmisión y la recepción de datos en los módulos de comunicaciones es el golpe fuerte por los cuales se produce una gran pérdida energética.

En una red de sensores, las medidas esporádicas consumirán menos energía que un control continuo. Además, niveles altos de ruido podrían causar una degradación significativa y aumentar la complejidad del algoritmo de detección, provocando un mayor gasto energético.

En cuanto a la energía requerida en el procesado de datos, hay que decir que es mucho menor comparada con la necesaria en el proceso de comunicación. De ahí, que sea crucial el procesado de datos local para reducir al mínimo el consumo de energía en una WSN.

Contando con que hay que tener buena eficiencia energética y que es crucial en redes de sensores inalámbricas, Waspmote, diseñado por la empresa Libelium, fue el resultado de 2 años de investigación. Se disminuyó el consumo de los dispositivos al mínimo, mientras que se aumentaban sus posibilidades de comunicación y de integración de nuevos sensores.

2.2. Dispositivos utilizados que engloba la plataforma Waspmote para el análisis

2.2.1. Funcionalidades de la plataforma Waspmote

Como ya se ha explicado anteriormente, Waspmote está compuesto por una serie de elementos necesarios para el correcto funcionamiento de la misma, de todos los sensores y módulos de comunicaciones que lo integran ya que está basado en una arquitectura modular y se requiere de ellos adaptándose a las necesidades de cada uno.

Cada funcionalidad conlleva una serie de funciones que son las que se ha analizado y que se desarrollaran más adelante. Las funcionalidades, que se explicarán con más detalle en el Anexo B de este documento, que se analizan son:

- Leds
- Memoria EEPROM
- Reloj de tiempo real o RTC
- Tarjeta de memoria micro-SD.
- Acelerómetro

- Sistemas de Energía:
  - Deep Sleep
  - Hibernate
  - Nivel de batería

2.2.2. *Placas integradas en la plataforma junto con la gama de sensores que forman cada una de ellas*

El dispositivo *Waspmote* dispone de 10 E/S digitales para poder conectar todo tipo de sensores externos, así como de 8 entradas analógicas. De esta forma obtenemos una gran cantidad de posibilidades de interconexión para nuestras redes sensoriales.

Se dispone de 10 placas de sensores. Cada una de ellas tienen una finalidad y permiten controlar múltiples parámetros en el medio ambiente involucrando una amplia gama de aplicaciones, dotándose de una gran variedad de sensores para tales fines y que se proceden a analizar (se ve más adelante). Son las siguientes:

- Placa de Agricultura: incluyen sensores como temperatura, humedad, presión, luminosidad, etc…
- Placa de Eventos: incluye sensores como piezoeléctrico, nivel de líquido, sensor de presencia PIR, efecto hall, etc…
- Placa de Gases: incluye sensores como oxígeno, monóxido de carbono, metano, dióxido de nitrógeno, etc…
- Placa de Smart Water: incluye sensores de conductividad, oxigeno disuelto, iones disueltos, etc…
- Placa de Smart Cities: incluye sensores como micrófono, ultrasonido, detección de rotura, partículas, etc…
- Placa de Parking: incluye el sensor de temperatura y el del campo magnético.
- Placa de Videocámara: incluye sensores como el de imagen, luminosidad, infrarrojo, etc…
- Placa de radiación: incluye un único sensor de radiación.
- Placa de Smart Metering: incluyen sensores de corriente, nivel de líquido, flujo de agua, carga pesada, etc…
- Placa de prototipado.

Fig. 2.1. Placa de Agricultura con sus sensores

2.2.3. *Módulos de comunicaciones*

Waspmote también tiene la capacidad de integrar una serie de módulos necesarios en cada dispositivo. Estos módulos se pueden cambiar y ampliarse según las necesidades. Cada módulo tiene múltiples utilidades y funciones a realizar que se desarrollan en el Anexo B.

Los módulos utilizados para las pruebas y que permiten integrarse en Waspmote son:

- XBee 802.15.4: con una banda de trabajo de 2.4 Ghz usando 12 canales con un ancho de banda de 5 MHz por canal. Las antenas que se pueden usar para la transmisión de datos son de 5 y 2 dBi.
- XBee 868: con una banda de frecuencia de 868 MHz usando 1 único canal con un ancho de banda de 0.5 MHz y una antena de 4.5 dBi.
- XBee 900: con una banda de frecuencia de 900 MHz usando 12 canales con un ancho de banda de 2.16 MHz y una antena de 4.5 dBi. Aunque este módulo también se integra en Waspmote, se ha descartado porque lo han dejado de vender.
- XBee DM (digiMesh): pueden compartir el módulo hardware con el 802.15.4 y con el 900. Nosotros lo hemos hecho con el 802.15.4 con lo cual tanto la banda de frencuencia, los canales, el ancho de banda y la antena serán los mismos que con el 802.
- XBee ZigBee: con una banda de frecuencia de 2.4 GHz, usando 14 canales y antenas de 2 y 5 dBi.
- Bluetooth: tiene múltiples acciones entre ellas descubrir otros dispositivos bluetooth. Usa 79 canales con un ancho de banda de 1 MHz por canal. Además, los saltos de frecuencia adaptable (AFH) se usan para mejorar las trasmisiones. La banda de frecuencia es de 2.4 GHz y la antena usada es de 2 dBi.
- BLE (bluetooth low energy): Usa una banda de frecuencia de 2.4 GHz. Tiene 37 canales de datos y 3 canales de avisos. Tiene múltiples acciones entre ellas está el

envío de avisos por Broadcast, descubrir otros dispositivos, etc.. Las antenas que se pueden usar son las de 2 y 5 dBi.

- Wifi: utilizando una banda de frecuencia de 2.4 GHz y posibilidad de usar antes de 2 y 5 dBi.
- GPRS: utilizando una antena externa de 0 dBi.
- GPS: utilizando una antena con un alcande de 26 dBi.
- 3G: utilizando una antena externa de 0 dBi.

2.3. Variables a medir

Sabemos que:

Energía=Potencia*Tiempo

Por tanto, para poder caracterizar el comportamiento energético del sistema monitorizaremos la tensión de la batería y la corriente demandada por el circuito alimentado utilizando un osciloscopio para conocer el perfil temporal de esas variables.

La autonomía de la batería se especifica en mA*h, siendo en el caso de nuestro producto el máximo admisible de 6600 mA*h. La tensión de la batería se comprobó que oscilaba entre 3.6V y 4.1V. Tras esa caracterización, se fijó un valor de tensión constante a 3.8V, correspondiendo a situaciones típicas de 50% de descarga de la batería.

Para la medida de corriente se empleó una sonda de corriente de modo que:

Potencia(t)=V(t)*I(t)

Así, en nuestra aplicación:

Energía=3.8V*I(t)*tiempo=[V]·[A]·[s]

Para hacer los cálculos más sencillos, sin necesidad de invertir mucho tiempo en el cálculo de la corriente consumida a lo largo del tiempo, se optó por hacer estimaciones aproximadas de las áreas de cada proceso a rectángulos, dividiéndose, cada estado, en secciones (ver apartado 5 de la memoria). Esta simplificación es correcta para la forma de onda de corriente con la que trabajamos.

La carga energética consumida en un evento de conexión, es el área que encierra ese rectángulo de altura en mA y anchura en ms. Por lo tanto, se calcula multiplicando esos dos términos:

Carga energética (mA*ms)= Ippromediototal(mA)*tiempo de ejecución(ms)

Por lo tanto, las variables que se va a medir en cada sección dividida van a ser la corriente de pico (Ip) y el tiempo t empleado en cada una de ellas, para, finalmente, calcular la corriente de pico promedio total y el correspondiente tiempo de ejecución durante ese proceso completo o evento de conexión.

2.4. Representaciones típicas

La caracterización del consumo se explica a continuación en base a representaciones típicas medidas como resultado de las pruebas realizadas. Las siguientes representaciones han sido uno de los resultados del trabajo:

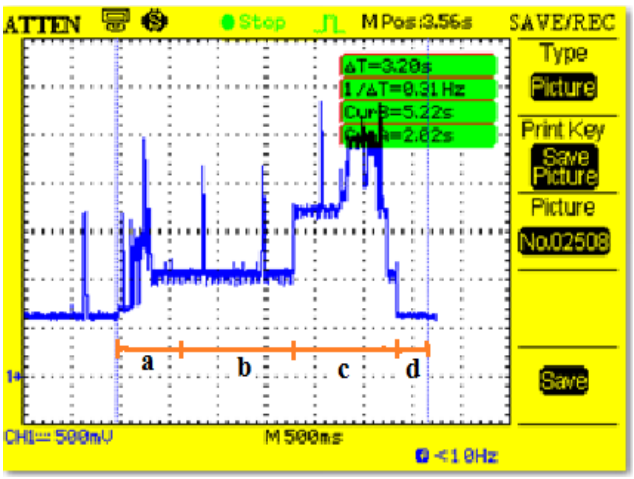


Fig. 2.1. Proceso completo de toma de una foto con placa de videocámara

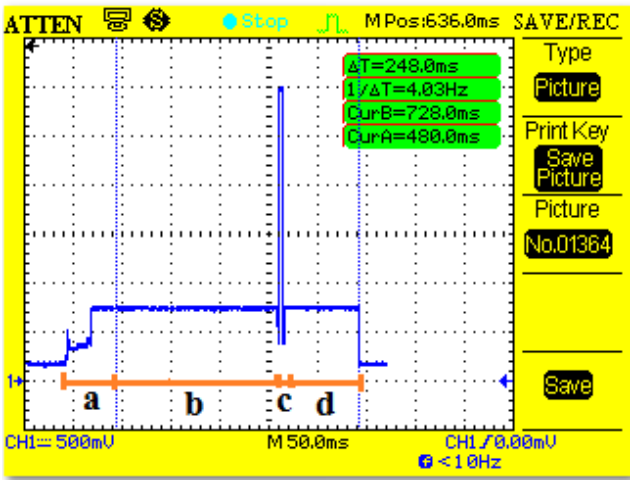


Fig. 2.2. Proceso completo de envío de datos por unicast a través del 802.15.4

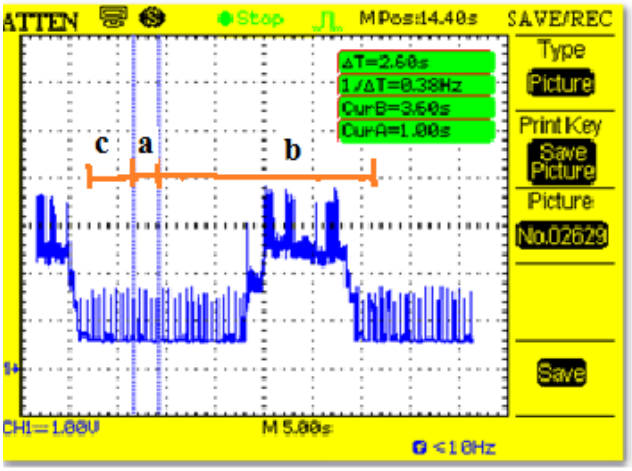


Fig. 2.3. Proceso completo de cómo subir un archivo a FTP desde Waspmote a través del 3G

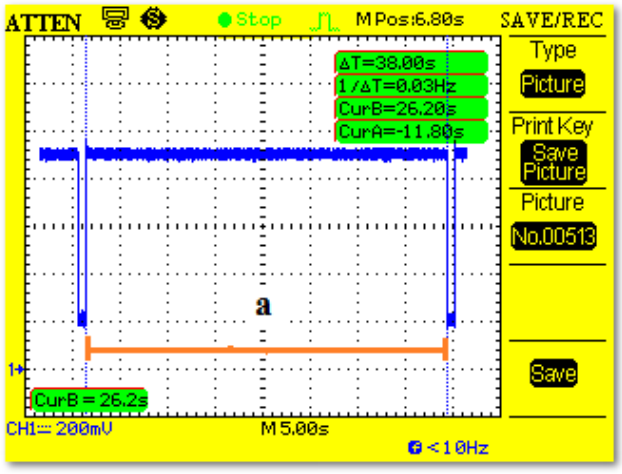


Fig. 2.4. Lectura sensor MiCS 2610 de gases

En esas figuras hemos podido identificar fases concretas relacionadas con el modo de trabajo de cada uno de los módulos participantes.

Así, como se ve en la fig. 2.1, el proceso de toma de una fotografía con la placa de videocámara está divido en 4 partes: proceso on de la cámara (a), configuración de la cámara (b), toma de foto (c) y proceso off cámara (d).

Como se ve en la fig. 2.2, el proceso está divido en 4 partes: proceso On del XBee 802.15.4 (a), pre-proceso al envío del paquete (b), envío del paquete (c) y post-proceso al envío del paquete (lectura del ACK) (d).

Como se ve en la fig. 2.3, el proceso está divido en 3 partes: configuración FTP (a), subida de archivos a FTP (b) y delay de 5 s (c).

Como se ve en la fig. 2.4, el proceso está divido en una única parte: proceso de lectura del sensor MiCS 2610 de la placa de gases (a).

3. Diseño de la configuración del sistema de medida.

En este apartado, se describe la instrumentación utilizada y la configuración general requerida para la realización de las pruebas de consumo.

3.1. Tecnología y herramientas empleadas

Durante el desarrollo del proyecto, se han empleado un tipo de lenguaje de programación y herramientas de desarrollo de los cuales, a continuación, se destacan:

- Para la programación de la API, se ha utilizado el entorno de desarrollo IDE, y el lenguaje de programación C++:
  - **Waspmote IDE**, como así se llama, es un entorno de desarrollo que ha sido empaquetado como un programa de aplicación. En él, se compila los códigos y desde donde se ejecutan. Aquí se incluyen las librerías y se trabajan con ellas.
  - **C++** es un lenguaje de programación orientado a objetos, utilizado para desarrollar la API.
- Para visualizar las gráficas y medidas de consumos energéticos se ha utilizado un **osciloscopio**, cuyo modelo es ADS2062CA de la serie ADS2000.

  Especificaciones:

  - Marca: Atten Intruments
  - BW: mHz 60
  - Max. La tasa de la muestra: 1gsa/s
  - Max. Profundidad de la memoria: 4k-1m/ch
  - Max. De forma de onda tasa de captura: 1000wfms/s
  - Canales: 2

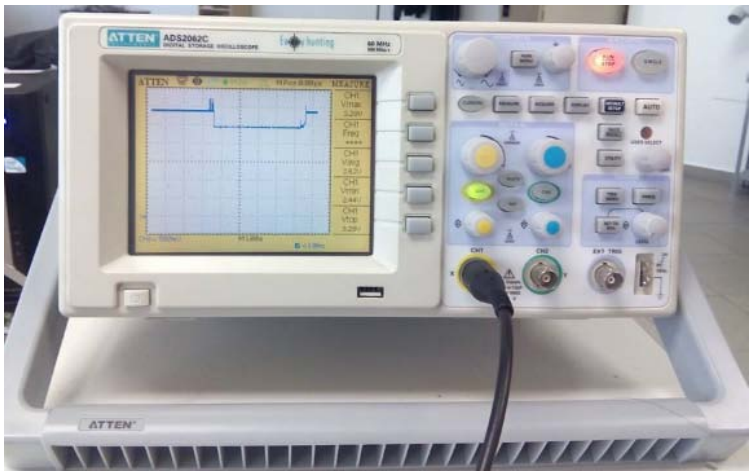


Fig. 3.1. Osciloscopio ADS2062CA

- Para medir la corriente se ha utilizado una **sonda amperimétrica AC/DC**, de 10mV/1mA y una Idc o de pico de 450mA, suficiente para nuestras mediciones. La marca de la sonda es CHAUVIN ARNOUX.

Fig. 3.2. Sonda Amperímetrica Chauvin Arnoux [10]

- Utilización de dispositivos hardware, siendo lo principal para las medidas: uso de la plataforma Waspmote ya descrita anteriormente con toda su gama de sensores y módulo de comunicaciones (se explican más a fondo en Anexo B).

3.2. Visión del conjunto del sistema

Con el fin de medir correctamente el consumo de corriente media, la corriente debe ser medida con respecto al tiempo. Por lo tanto, un multímetro de base no es suficiente, y se requiere un osciloscopio para visualizar la medida realizada.

La forma más sencilla de medir la corriente con un osciloscopio, es utilizar una sonda de corriente y directamente controlar la corriente que va al sistema. Una alternativa es utilizar una pequeña resistencia en línea con la entrada de alimentación al sistema. A continuación, se utilizaría una sonda de voltaje del osciloscopio estándar, se mediría la tensión en la resistencia y, así, calcular la corriente dividiendo el voltaje por el valor de la resistencia. Un buen valor de resistencia a utilizar sería de 10Ω. Este valor es lo suficientemente pequeño para que no afecte a los circuitos existentes, y lo suficientemente grande para proporcionar una tensión que se puede medir con una decente precisión. Además, el uso de un valor de 10Ω hace los cálculos muy fáciles. Como la empresa dispone de sonda amperimétrica, puedo medir la corriente a través de su pinza de corriente sin necesidad de colocar la resistencia.

En algunas ocasiones, el consumo de algún dispositivo es de órdenes de muy pocos uA que con el osciloscopio es imposible de medir. Para ello, se utilizó un polímetro que es más preciso y puede medir corrientes de órdenes de uA.

Para realizar las mediciones, se ha utilizado como fuente de alimentación una batería de litio ICR18650H-1S3P recargable cuya tensión varía entre 3.7V y 4.1V, con una capacidad de 6600mA*h, aunque se recomienda utilizar una fuente de alimentación de CC regulada y fija en lugar de una batería real, por no estar recargando la batería constantemente.

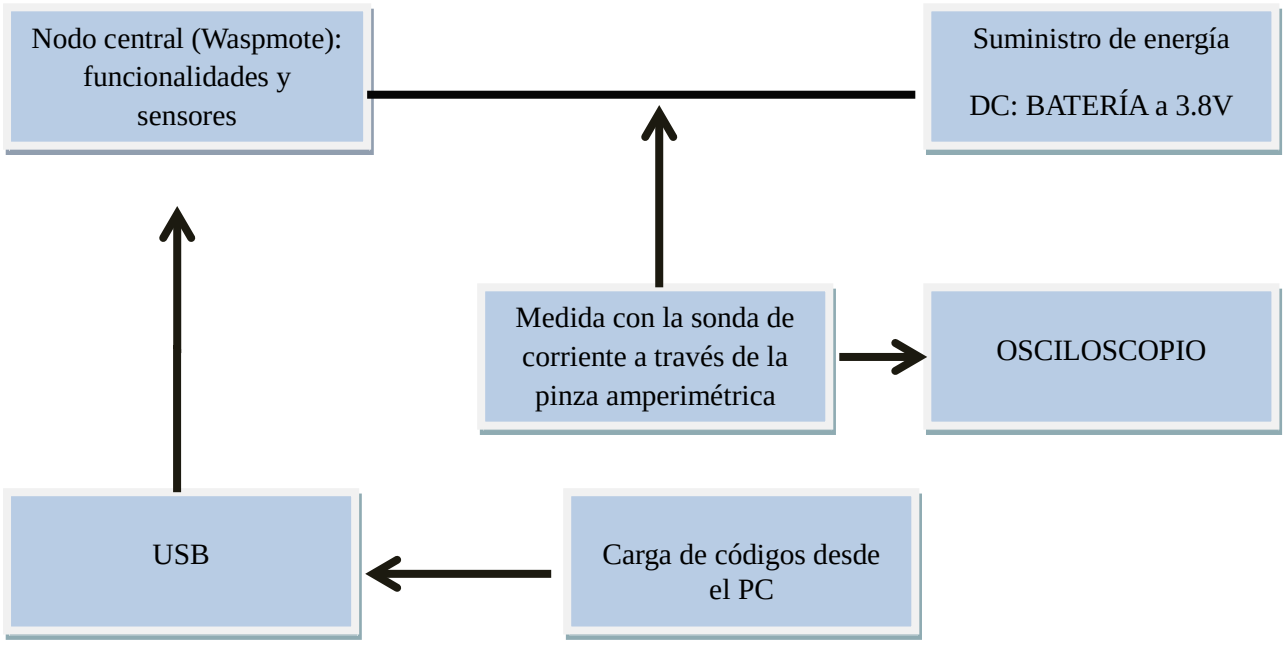
Se ha comprobado que las baterías seleccionadas no sean defectuosas ya que esto podría no eliminar variables y afectaría a la medida y que no sea baja, es decir, que no esté descargada porque aunque no difiera mucho en el consumo, se debe evitar que sea baja para tener una medida más real, con lo cual, se eligió y se tomó por buena una tensión media de 3.8V, que se corresponde al 50-60% de carga de la batería.

Fig. 3.3. Batería de litio recargable

Respecto al software, para medir el consumo de cada uno de los códigos ejecutados, se cargarán a través del USB que se conecta desde el ordenador al Waspmote.

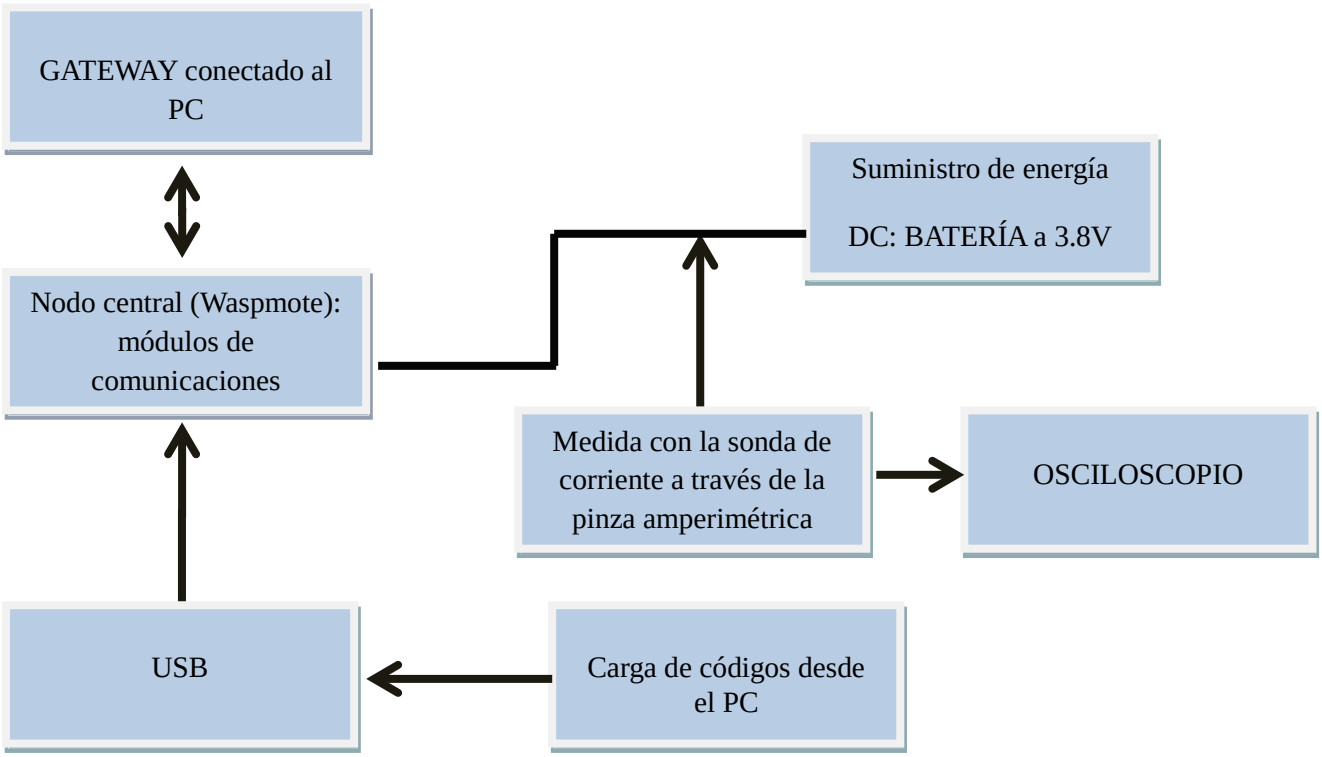
Los siguientes esquemas muestran la configuración completa seguida:

Primer esquema: Utilizando el nodo central (Waspmote) para medir sus diferentes modos de funcionamiento y los sensores que se integran en él.



Los códigos utilizados se cargaran utilizando el compilador IDE desde el ordenador a través del USB hasta el nodo central, que será el Waspmote. La medida se realizará a través de una sonda de corriente puesta en la batería que se recogerá y visualizará en el osciloscopio.

Segundo esquema: Utilizando el nodo central (Waspmote) para medir los diferentes módulos de comunicaciones.



Los códigos utilizados se cargaran utilizando el compilador IDE desde el ordenador a través del USB hasta el nodo central, que será el Waspmote. La trasmisión de los paquetes se llevará a cabo a través de un segundo módulo, funcionando como receptor colocado en el Gateway que a su vez se conectará al ordenador: funciona como un USB. La medida se realizará a través de una sonda de corriente puesta en la batería que se recogerá y visualizará en el osciloscopio.

Las siguientes figuras, muestran, visualmente, la configuración del sistema de medida:

Fig. 3.4. Setup de medida midiendo con el módulo 3G

Fig. 3.5. Zoom del setup de medida con el módulo 3G

Fig. 3.6. Setup de medida con un módulo ZigBee

En la primera, se muestra un ejemplo de cómo se mediría, en este caso, con un módulo 3G integrado en el Waspmote. En este caso no necesita Gateway ya que las funciones que realiza este tipo de módulos son distintas a la de que haría por ejemplo el 802.15.4, 868, DM o el ZigBee.

La segunda es un Zoom de la primera, mostrando más cercanos los dispositivos tanto de la plataforma Waspmote, el módulo 3G, la batería y la sonda de corriente.

La tercera imagen, muestra como sería una medida con el módulo ZigBee. En este caso, se ve que se necesita un Gateway con otro módulo para la transmisión de los paquetes. En este caso, el Coordinador es el del Gateway y el router el del Waspmote.

3.3. Hardware adicional utilizado

Waspmote Gateway [11]:

Se utiliza principalmente para la transmisión y recepción de datos de paquetes a través de tramas en los módulos de comunicaciones.

Este dispositivo permite recoger datos que fluye a través de la red de sensores en un PC o dispositivo con un puerto USB estándar.

El Waspmote gateway actúa como "Puente de datos o punto de acceso" entre la red de sensores y del equipo de recepción.

Este equipo receptor será responsable de almacenar y utilizar los datos recibidos en función de las necesidades específicas de la aplicación.

El equipo receptor puede ser un PC con Linux, Windows o Mac OS, o cualquier dispositivo compatible con conectividad USB estándar.

Una vez que la puerta de enlace se ha instalado correctamente, un nuevo puerto serie de comunicación que conecta directamente a la UART del módulo XBee aparece en el equipo receptor, permitiendo que el XBee se comunique directamente con el dispositivo, pudiendo ambos recibir paquetes de datos de la red de sensores, así como modificar y / o consultar los parámetros de configuración del XBee.

Otra función importante de destacar es la posibilidad de actualizar o cambiar el firmware del módulo XBee.

Fig. 3.7. Waspmote Gateway [11]

3.4. Modificaciones de Hardware

Para que la medida realizada, se parezca a la realidad, se hizo una pequeña modificación en el hardware del Waspmote.

Se estudió, en un principio, 2 posibilidades para realizar las medidas:

- Primera: evitar hacer modificaciones del Hardware. Esto es, primeramente, cargar el código a través del USB y posteriormente, para realizar las medidas con la sonda, quitar el USB del PC, así evitar un consumo extra producido por el USB.
- Segunda: hacer modificaciones de Hardware. El USB, a través del cual, se cargan los códigos que vamos a medir y caracterizar, se alimenta a una tensión de 5V con una corriente de 100 mA. Esto va a modificar y falsear completamente la medida, sacando más de lo que en realidad es, ya que tendríamos en cuenta esa corriente, pudiéndonos hacernos creer que la batería no consume. Con lo cual, para evitar que pasen esos 100 mA extra y hacer que la medida sea lo más real posible, se hizo un corte de pista. Así, se evita que la medida no esté falseada y, así, hacer que la alimentación, sólo proceda de la batería.

Se optó por el segundo punto, para evitar que se quite y se ponga todo el rato el cable USB y hacerlo de esta manera más cómoda. Pero, esto acarrea un problema, y es que al no suministrar corriente que suministraba el USB, la batería ya no se recargará. Con lo cual, empezará a descargarse poco a poco, acortando su tiempo de vida dependiendo del proceso que se quiere ejecutar. Esto es bueno para el usuario, porque es lo que realmente se persigue, que no interfiera nada para ver cuando aguanta a batería hasta descargarse, pero para mí, que es la que está midiendo todo el rato, se hace un poco molesto porque cada cierto tiempo, para mantener el nivel de batería a 3.8V (50/60 % de batería), tengo que ir recargándola.

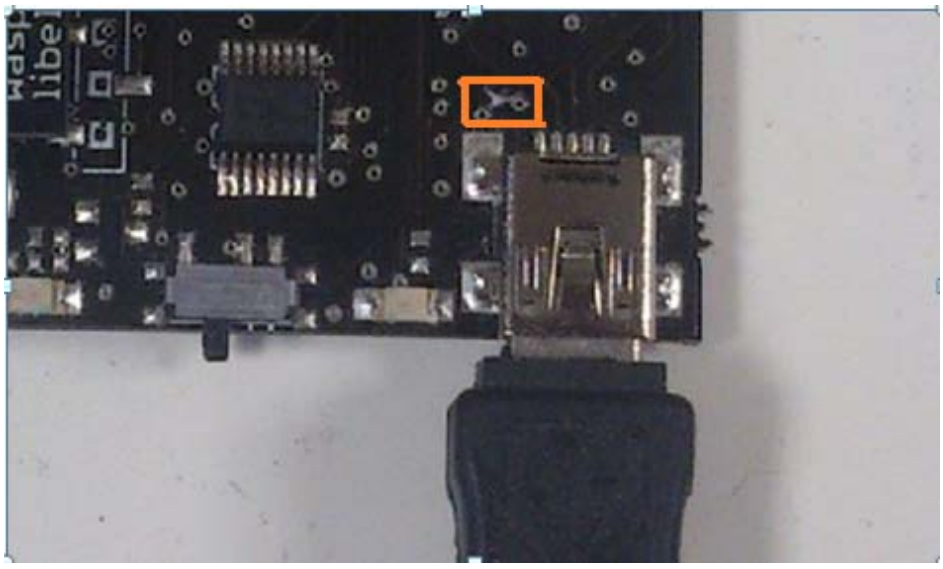


Fig. 3.8. USB de Waspmote con la pista cortada

3.5. Software Setup

En este apartado se va a describir como se ha configurado el software de medida en los tres grupos (funcionalidades, sensores y módulos), utilizado para la caracterización de las medidas de consumo y obtenidas según el código que se ejecuta a través del programa de compilación, explicado anteriormente.

Lo primero que hay que saber es que la gran plantilla que forma la empresa Libelium, ha sido la que ha diseñado e implementado cada uno de los códigos que se ha utilizado consiguiendo que realicen las acciones pertinentes de cada función que hace cada dispositivo utilizado y que yo he modificado ligeramente (véase el siguiente apartado).

3.5.1. *Programación del Software.*

La estructura de los códigos del ".Pde" utilizados se divide en 3 grupos básicos:

- Primero: donde se incluyen las librerías utilizadas en la API de Waspmote, definiciones y inicializa el código (o el Waspmote se reinicia). Aquí se incluyen la inicialización tanto de la placa de sensores como de los módulos que se van a utilizar, así como la parte del código que sólo es importante cuando se inicia Waspmote.
- Tercero: el loop(). Es la segunda parte del código. Se ejecuta continuamente, formando un bucle infinito. El objetivo de esta función es medir, enviar la información y el ahorro de energía mediante la introducción de un estado de bajo consumo. Esta última, se ha obviado, porque, como se explica más adelante, los códigos ejecutados se han hecho sin llevar el Waspmote a modo Sleep.

A continuación, se muestra la estructura general del programa que se ha seguido, separada en grupos de un ejemplo con un módulo:

```
// 1. Include Libraries
// 2. Definitions
// 3. Global variables declaration

void setup()
{
   // 4. Modules initialization
}

void loop()
{
   // 5. Measure
   // 6. Send information
   // 7. Sleep Waspmote
}
```

Fig. 3.9. Estructura seguida de un ejemplo con un módulo [12]

Si se le echa un vistazo a los cientos de ejemplos implementados, veremos que suelen cumplir este orden.

3.5.2. *Estructura software seguida en cada grupo*

En este apartado se va a mostrar con ejemplos reales compilados, la estructura que se ha seguido en cada grupo. Será diferente de acuerdo a lo que se ejecute.

Funcionalidades internas del Waspmote:

Aquí no se requiere definir librerías. Se declararán variables y se procederá a ejecutará las funciones según el código que se compile.

A continuación, se muestran dos ejemplos diferentes:

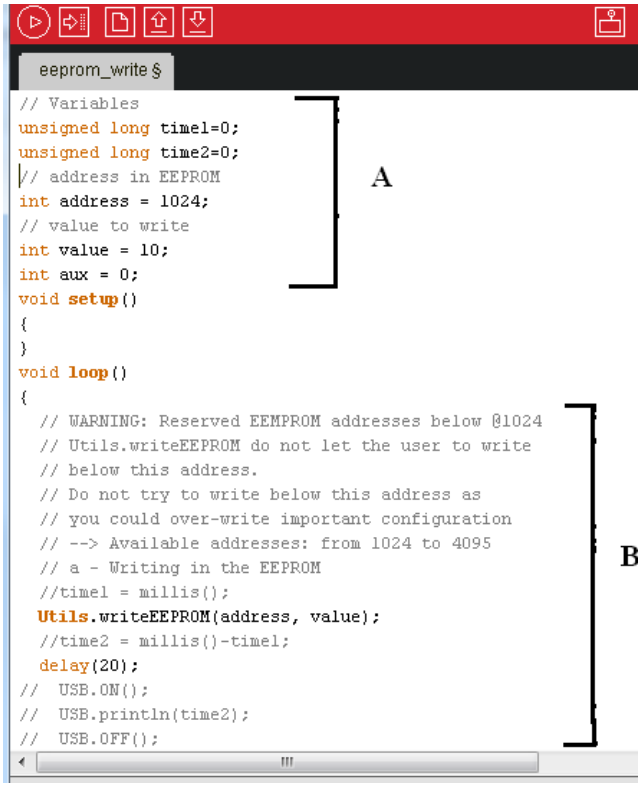
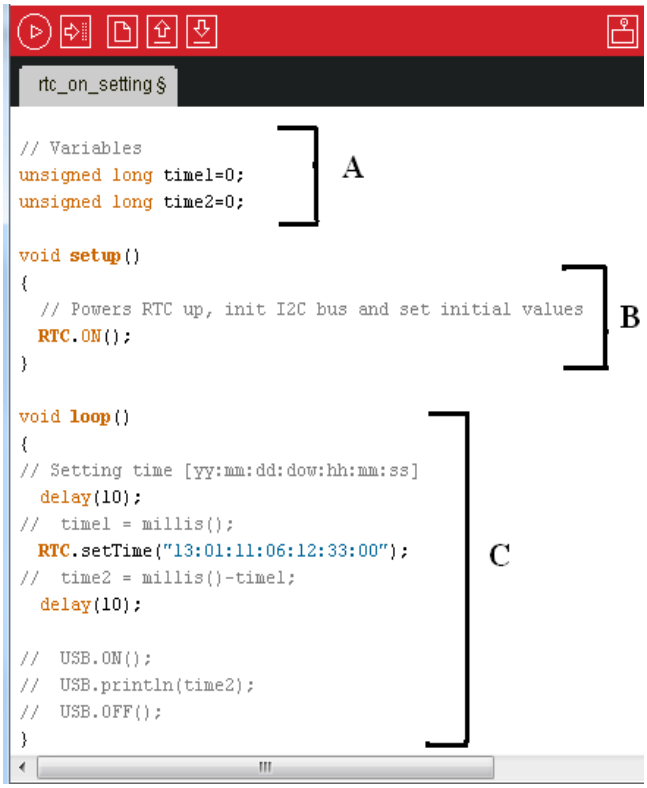


Fig. 3.10. Escritura en la memoria



3.11. Fijar fecha y hora del RTC

En el primer ejemplo se observa que para realizar la lectura de la EEPROM, no se necesita inicializar previamente nada en el setup(). Se procederá únicamente a declarar las variables (A) que se vayan a utilizar y se ejecutar el código para tal fin en el programa principal (B).

En el segundo ejemplo, es distinto ya que para fijar la fecha y hora del RTC ejecutándose en el programa principal (C), previamente se necesita encender el RTC, con lo cual se deja inicializado en el setup() (B). En este caso, no necesita declarar ningún variable que forme parte de la función ejecutada. Simplemente, se declaran las variables para medir el tiempo de ejecución de la función (A).

Sensores:

Aquí sí que se necesita definir las librerías correspondientes a la placa utilizada. Se declararán variables y se inicializarán las placas en el setup () para la lectura de los sensores integrados en cada una en el programa principal.

Comentar que, para ejecutar las funciones de encendido y apagado de cada placa, no se necesita declarar ningún tipo de variable. Simplemente se ejecutarán las funciones en el programa principal.

A continuación, se muestra un ejemplo:

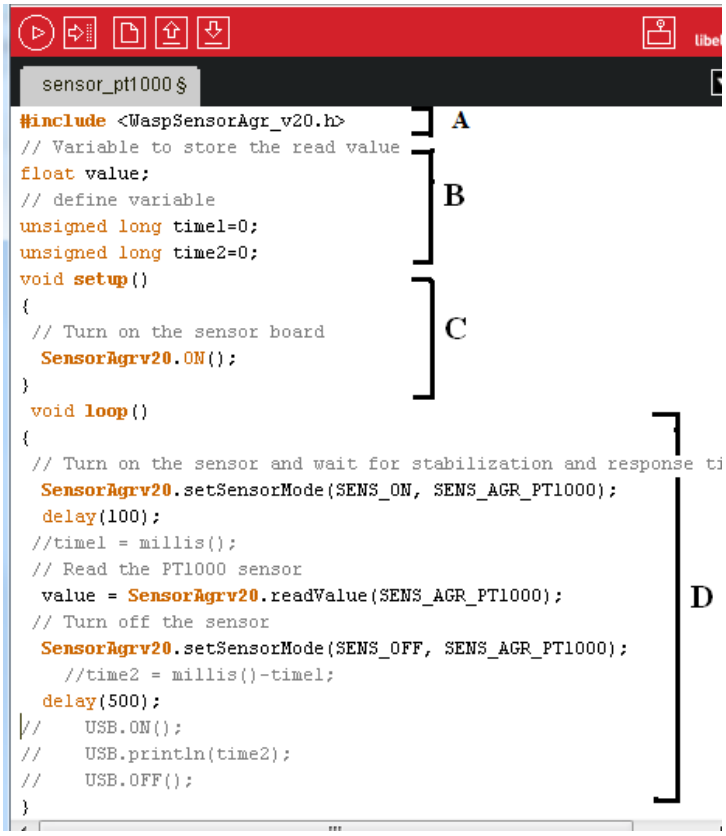


Fig. 3.12. Lectura sensor Pt1000 de la placa Agricultura

Se observa que se incluyen la librería (A), variables declaradas (B), inicialización previa de la placa de agricultura (C) y ejecución del código en el programa principal (D), en este caso, medición del sensor.

Cabe comentar que, en algunas placas de sensores como pasa con la de Eventos o la de Smart Water, no es necesario encender el sensor previamente para la lectura del mismo ya que su diseño funciona de diferente modo que los demás.

Módulos de comunicaciones:

Aquí, también se necesita definir las librerías correspondientes al módulo utilizado. Se declararán variables y a modo general, se inicializarán los módulos en el setup (), aunque en alguno,

por ciertas razones (se explica en la prueba correspondiente en el apartado C de los anexos), no seguirá esta estructura.

Comentar que para ejecutar las funciones de encendido y apagado de cada módulo, no se necesita declarar ningún tipo de variable. Simplemente se ejecutarán las funciones en el programa principal.

A continuación, se muestra un ejemplo:

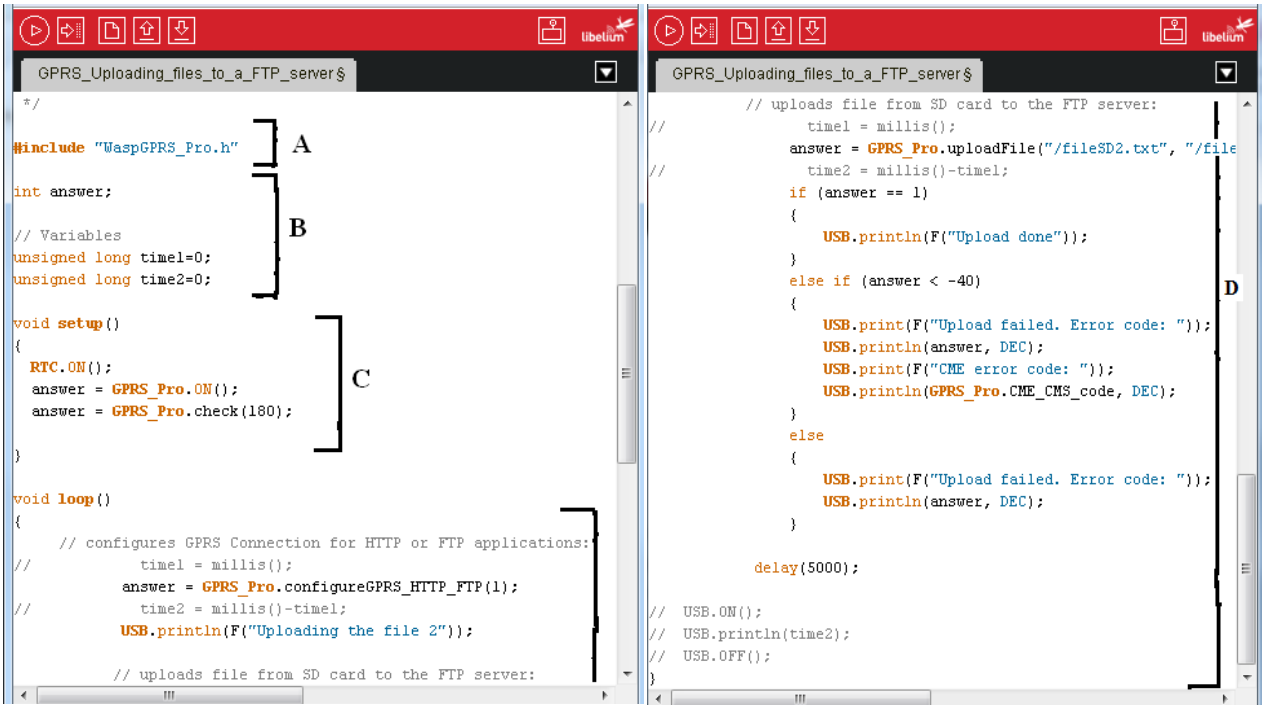


Fig. 3.13. Subida de archivos a FTP a través del módulo GPRS

Para este ejemplo, se observa que se incluyen la librería (A), variables declaradas (B), la inicialización previa del módulo GRPS y conexión a la red (C) y ejecución del código en el programa principal (D), en este caso, envío de información.

3.6. Software adicional utilizado

Serial Monitor [13]**:**

Es una herramienta usada por el compilador IDE donde se muestran los datos en serie que se envían desde la placa Waspmote (a través de USB o tarjeta serie).

Hay que seleccionar la velocidad de transmisión desde el desplegable que coincida con la tasa de Waspmote v1.2 (USB = 115200 bps) y el correcto puerto de salida.

Para enviar los datos a la placa, se escribe el texto y se hace clic en el botón "enviar" o se presiona *enter*.

Los datos recibidos por el Gateway a través del Serial Monitor eligiendo el correcto puerto de salida al que está conectado.

A continuación, se muestra un pantallazo de un ejemplo medido donde se muestran las tramas enviadas junto con el tamaño que ocupan desde un módulo Wifi, a su vez, se muestra los datos de encendido y apagado del mismo:

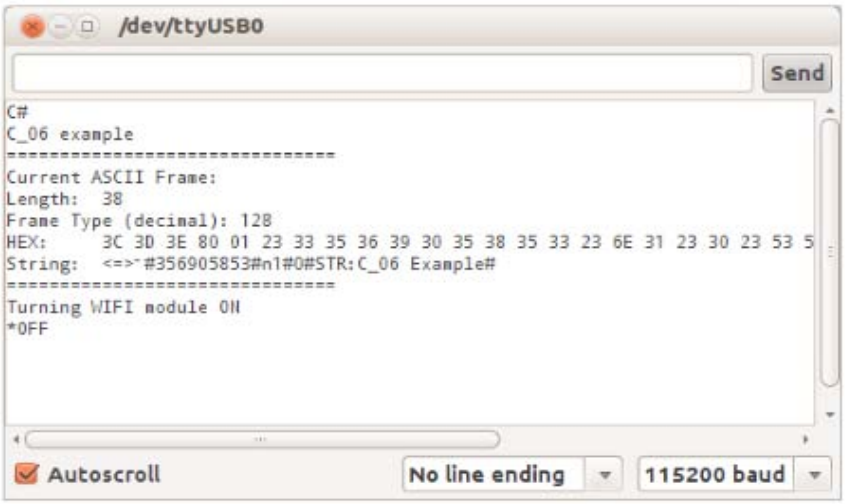


Fig. 3.14. Serial monitor con datos enviados desde módulo Wifi [13]

CuteCom:

Programa destinado a la visualización de datos enviados en serie desde la placa Waspmote. Tiene la misma finalidad que el Serial Monitor.

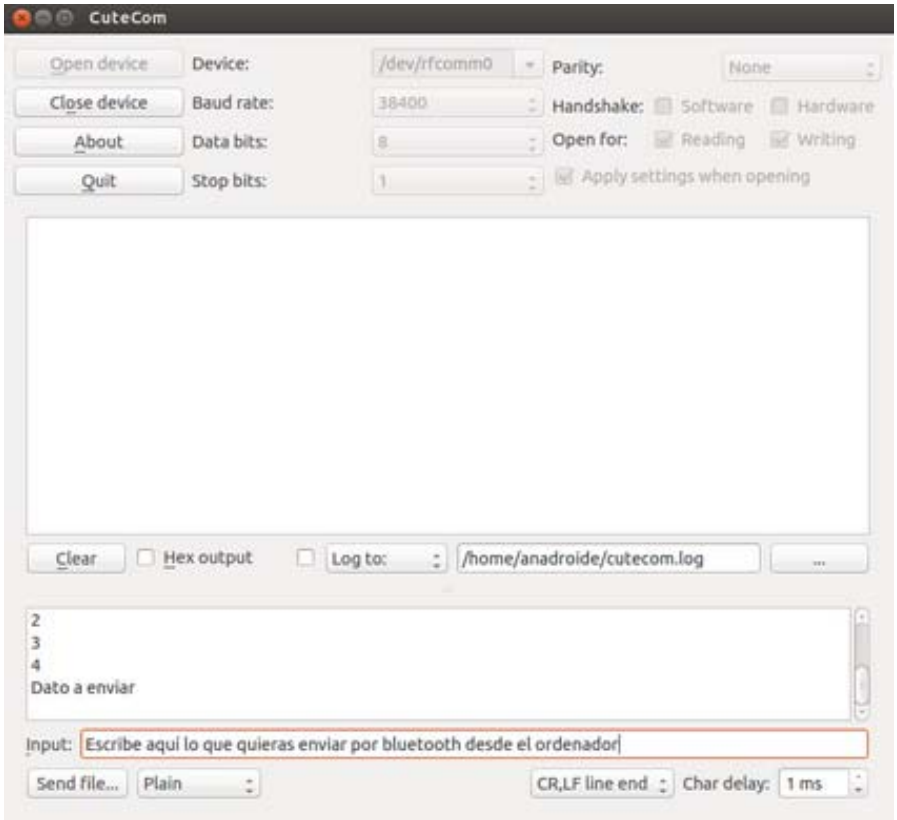


Fig. 3.15. Programa CuteCom [14]

Para que lleguen los datos a este terminal, previamente se tiene que abrir el dispositivo, presionando el botón *Open Device* y seleccionar el correcto puerto de salida.

Para enviar los datos a la placa, se escribe el texto y se presiona *enter*. Si se quiere borrar datos, se presiona el botón *clear*.

A continuación, se muestran los datos de los paquetes que se han enviado a través de un módulo ZigBee visualizándolos en el CuteCom:

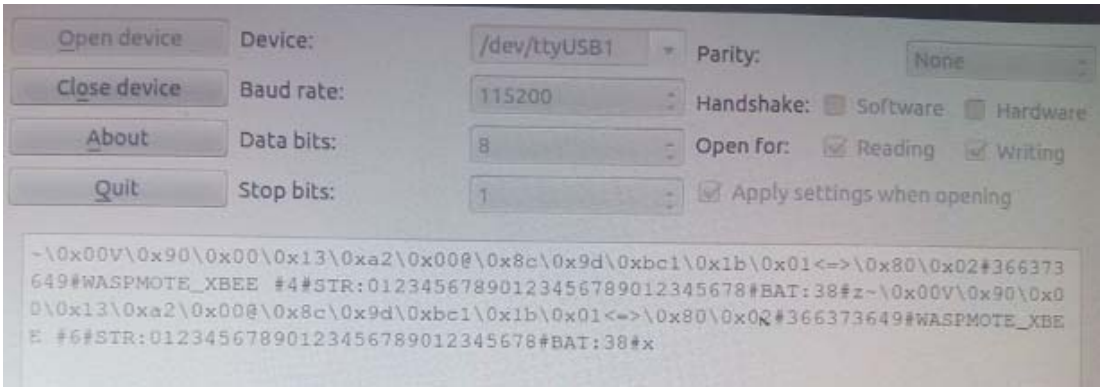


Fig. 3.16. Visualización de datos enviados con el CuteCom

3.7. Modificaciones del Software

El código se modificará ligeramente ya que tiene que ser lo más limpio y eficiente que se pueda, eliminando las acciones no necesarias que añadirían consumo irreal, es decir, las funciones de USB.ON o USB.OFF o USB.PRINTLN(). Estas, se comentarán en el código y así, se consigue no falsear la medida.

Estas funciones son muy útiles para dar información valiosa (como se ha visto en la imagen del apartado anterior) visualizándose en la pantalla del Serial Monitor del compilador IDE. Esta información diferirá una de otra dependiendo de la acción que se ejecute en el código. Por lo tanto, primeramente se ejecuta el código completo teniendo en cuenta estas funciones para comprobar que funcionan correctamente y que sale lo deseado, y una vez hecho esto, finalmente, se eliminarán para la realización de las pruebas de medida.

A continuación, se muestra un ejemplo de lectura de un sensor de la placa de Gases con los *USB. println*:

```
  // Turn on the sensor on socket 2A and wait for stabilization and
  // sensor response time
  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET2A);
  delay(40000);


}

void loop()
{

  // Read the sensor
  socket2AVal = SensorGasv20.readValue(SENS_SOCKET2A);

  // Print the result in volts and ohms through the USB
  USB.print(F("SOCKET2A: "));
  USB.print(socket2AVal);
  USB.print(F("V - "));

  // Conversion from voltage into kiloohms
  socket2AVal = SensorGasv20.calculateResistance(SENS_SOCKET2A, socket2AVal, GAIN,
RESISTOR);
  USB.print(socket2AVal);
  USB.println("kohms");

  delay(1000);
```

Fig. 3.17. Lectura sensor de Gases colocado en el socket 2A de la placa

Como se ve en el código, estos *USB.print* tienen la función de imprimir el valor del sensor de Gas en voltios y en kohms, de acuerdo a las acciones ejecutadas.

A continuación, se muestra como quedaría el código final eliminando estas funciones ya que añadirían consumo irreal:

```
#include <WaspSensorGas_v20.h>

#define GAIN  1      //GAIN of the sensor stage
#define RESISTOR 20  //LOAD RESISTOR of the sensor stage

//Variable to store the read value
float socket2AVal;

// Variables
unsigned long time1=0;
unsigned long time2=0;

void setup()
{
// Turn on the sensor board
  SensorGasv20.ON();
 / Turn on the RTC
  RTC.ON();
// Configure the 2A sensor socket
  SensorGasv20.configureSensor(SENS_SOCKET2A, GAIN, RESISTOR);
}
```

Fig. 3.18. Definición de las variables y configuración de las funciones para la lectura del sensor de gas

```
void loop()
{
//time1 = millis();
  // Turn on the sensor on socket 2A and wait for stabilization and
  // sensor response time
  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET2A);
  delay(40000);
   // Read the sensor
  socket2AVal = SensorGasv20.readValue(SENS_SOCKET2A);
  socket2AVal = SensorGasv20.calculateResistance(SENS_SOCKET2A, socket2AVa
  //Turn off the sensor on socket 2A
  SensorGasv20.setSensorMode(SENS_OFF, SENS_SOCKET2A);
  delay(1000);
  //time2 = millis()-time1;


//  USB.ON();
//  USB.println(time2);
//  USB.OFF();
}
```

Fig. 3.19. Programa principal de cómo sería leer un sensor de gas

En algunas ocasiones y dependiendo del dispositivo utilizado para la realización de las medidas, hace falta dejar este tipo de funciones, debido a que el dispositivo, como ocurre con según que módulos, por lo que sea, da errores, la mayoría de las veces, en el proceso de ejecución, por lo tanto, para asegurar que se ejecutan correctamente, es preferible dejarlos.

Además, en los módulos de comunicaciones, el tiempo de ejecución de las acciones es grande y su consumo también, por lo que no modificaría en su consumo dejar estas funciones en el código ya que si lo comparamos, la diferencia es mínima.

A continuación se muestra un ejemplo de ello con el módulo 3G:

```
_3G_Uploading_files_to_a_FTP_server_from_3g_module_SD §

#include "Wasp3G.h"

int answer;

// Variables
unsigned long time1=0;
unsigned long time2=0;


void setup()
{
  answer = _3G.ON();
  answer = _3G.check(60);
}

void loop()
{
    // configures onnection for FTP applications:
//            time1 = millis();
            answer = _3G.configureFTP("212.142.132.78", "21", "t3g@libelium.
//            time2 = millis()-time1;
            if (answer == 1)
            {
                USB.println(F("Uploading the file..."));
```

```
_3G_Uploading_files_to_a_FTP_server_from_3g_module_SD §

                // uploads file from 3G module (D:/Picture directory) to t
//                time1 = millis();
                answer = _3G.uploadFile(4, "19800106_000157.jpg");
//                time2 = millis()-time1;
                if (answer == 1)
                {
                    USB.println(F("Upload done"));
                }
                else if(answer == -2)
                {
                    USB.print(F("Upload failed. Error code: "));
                    USB.println(answer, DEC);
                    USB.print(F("CME error code: "));
                    USB.println(_3G.CME_CMS_code, DEC);
                }
                else
                {
                    USB.print(F("Upload failed. Error code: "));
                    USB.println(answer, DEC);
                }
            }
            else
            {
                USB.println(F("Configuration failed. Error code:"));
                USB.println(answer, DEC);
            }
```

Fig. 3.20. Ejemplo “Subir un archivo de una imagen desde el módulo 3G al servidor FTP”

En este ejemplo en concreto, es importante dejar este tipo de funciones porque en algunas ocasiones dan errores en la subida de archivos, y los USB. println dan información de si se ha subido correctamente o en caso negativo, te dicen el tipo de error que se ha producido para que se corrija.

Cabe destacar que, en mi proyecto, los códigos se han ejecutado con el Waspmote permanentemente encendido y no mandándolo a Sleep y, así, saber el consumo de cada proceso ejecutado por separado.

Para calcular más exactos el tiempo que tarda en ejecutarse cada acción y así poder señalar en las medidas, hasta donde llega cada proceso, en el código se añade una serie de funciones para tal fin. Cabe destacar, que estas funciones se ejecutan antes de analizar las medidas, ya como se ha dicho anteriormente, añadirían consumo irreal. Por lo tanto, para el diseño final, se dejan comentadas también.

Para verlo tomamos un ejemplo sencillo del encendido (como el que se ve a la derecha) y apagado del LED 0 de la plataforma Waspmote:

Como se ve, se definen las variables *time 1* y *time 2* antes del *setup*. Las funciones *time1 = millis()* y *time2 = millis()-time1* se ponen entre la acción que se quiera medir y son las que sirven para calcular el tiempo en ms y, a través de la función *USB.pinrtln(time2)*, se saca el valor visualizándose en el Serial Monitor del compilador IDE.

```
LED0_on_off§
 // Variables
unsigned long time1=0;
unsigned long time2=0;

void setup()
{
}

void loop()
{
  ////////////////////////////
  // 1. Setting LEDs ON
  ////////////////////////////
 //time1 = millis();
  Utils.setLED(LED0, LED_ON);
 //time2 = millis()-time1;
  delay(1000);
  ////////////////////////////
  // 2. Setting LED0 OFF
  ////////////////////////////
  Utils.setLED(LED0, LED_OFF);
  delay(1000);

//  USB.ON();
//  USB.println(time2);
//  USB.OFF();
}
```

Fig. 3.21. Proceso On y Off del Led 0

Para ver que esto tiene gran utilidad, se va a mostrar una gráfica como resultado de la prueba realizada distinguiendo los diferentes procesos que ocurren de acuerdo al código ejecutado:

```
DM_04a_send_broadcast§
////////////////////////////////////////////
// 1. Create ASCII frame
////////////////////////////////////////////
// 1.1. create new frame
frame.createFrame(ASCII, "WASPMOTE_04a");
// 1.2. add frame fields
frame.addSensor(SENSOR_STR, "01234567890123456789012345678901234567890123456");//37
//frame.showFrame();
////////////////////////////////////////////
// 2. Send packet
////////////////////////////////////////////
xbeeDM.ON();                                                              a
delay(100);                                                               b
// 2.1. set parameters to packet:
packet=(packetXBee*) calloc(1,sizeof(packetXBee)); // memory allocation
packet->mode=BROADCAST; // set Unicast mode
// 2.2. set destination XBee parameters to packet
xbeeDM.setDestinationParams(packet, "000000000000FFFF", frame.buffer, frame.length);
// 2.3. send data                                                         c
//time1 = millis();
xbeeDM.sendXBee(packet);
//time2 = millis()-time1;
// 2.4. free memory
free(packet);
packet=NULL;
xbeeDM.OFF();
delay(1000);                                                              d
```

Fig. 3.22. Código envío de paquetes por broadcast y cifrado a través del módulo DM

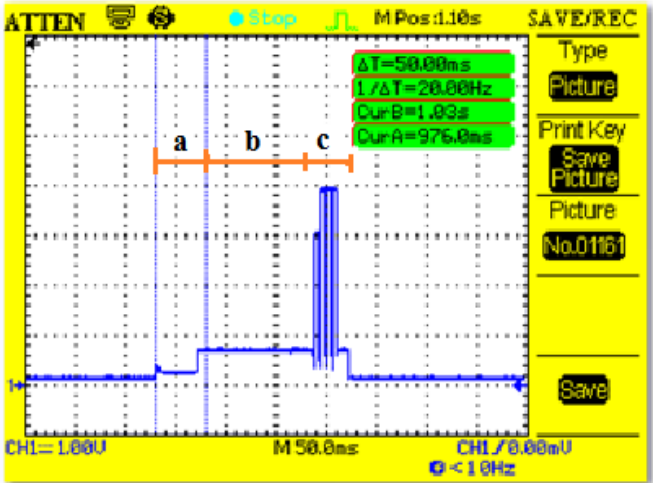


Fig. 3.23.a. Envío de los paquetes dividido en partes

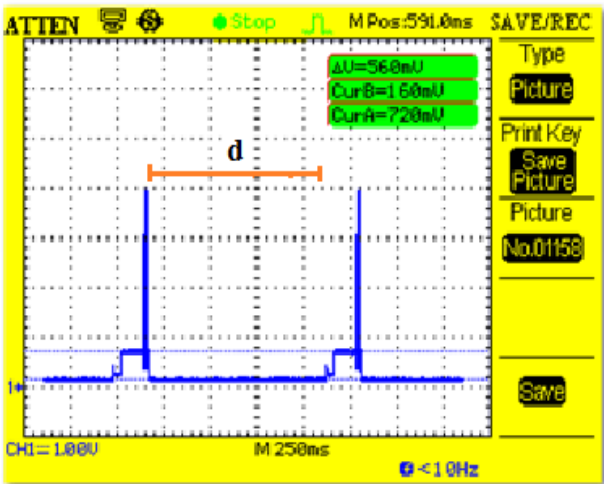


Fig. 3.23.b. Medida general cada 1000 ms

Este ejemplo corresponde al envío de datos de paquetes por Broadcast y en modo cifrado a través del módulo DM. Como se ve, se ha dividido el código en 4 partes. La inicialización del módulo (a), que conlleva tiempo de ejecución, el delay de espera de 100 ms (b), todo el proceso de envío de paquete (c) y el delay al final después de apagar el módulo, para realizar el proceso para 1000 ms (d). Como se ve, el proceso a Off del módulo es instantáneo, por eso no se ha marcado nada.

Cabe comentar que, aunque sepamos, aplicando la función `millis`(), el tiempo de ejecución

de cada acción, no sabemos en qué punto empieza y acaba cada una de ellas, por lo tanto, para distinguirlo, se añaden una serie de funciones antes y después de la acción que se ejecuta. Estas funciones lo que hacen es poner forzar un consumo extra al proceso que se está midiendo pudiéndose ser observado con facilidad en el osciloscopio y facilitar mejor la distinción entre dónde y dónde va a durar cada acción (se irá viendo en el apartado 5 de la memoria y en los ejemplos del Anexo C).

4. Medidas y análisis.

En este apartado, se va a explicar, cogiendo un modelo de ejemplo medido, el modo que se ha seguido para caracterizar las medidas realizadas y cuáles han sido los pasos para analizarlas junto con el cálculo del consumo en cada una de ellas.

Cabe destacar que este modo de caracterizar y analizar medidas seguido en la empresa Libelium, coincide con otras empresas utilizando el mismo proceso, por ejemplo **Texas Intruments**.

Dada la gran cantidad de medidas realizadas y de los datos obtenidos, se dedica este apartado a exponer con una serie de ejemplos la técnica empleada para el cálculo del consumo de un ejemplo global para estimar la vida de la batería del dispositivo, es decir desde que se despierta Waspmote hasta que se desactiva. Estos ejemplos permitirán entender la técnica desarrollada y aplicada para obtener los resultados generales que han permitido elaborar la Guía Energética.

Este método se ha aplicado también para el cálculo del consumo de cada ejemplo que se ha medido, con la diferencia que, en vez de tomar como referencia el consumo desde los 55uA en modo Sleep, se toma el correspondiente al dispositivo conectado en ese momento.

Tomando como ejemplo la medida del consumo en un evento de conexión del módulo de comunicación 802.15.4, se va a explicar cómo hay que analizar la forma de onda calculando el consumo para cada estado que se produzca en todo el proceso para finalmente saber el consumo total de todo el evento de conexión y así, calcular la vida de la batería que supondría ejecutar tal proceso.

La forma de onda de la medida global del evento de conexión es la siguiente:

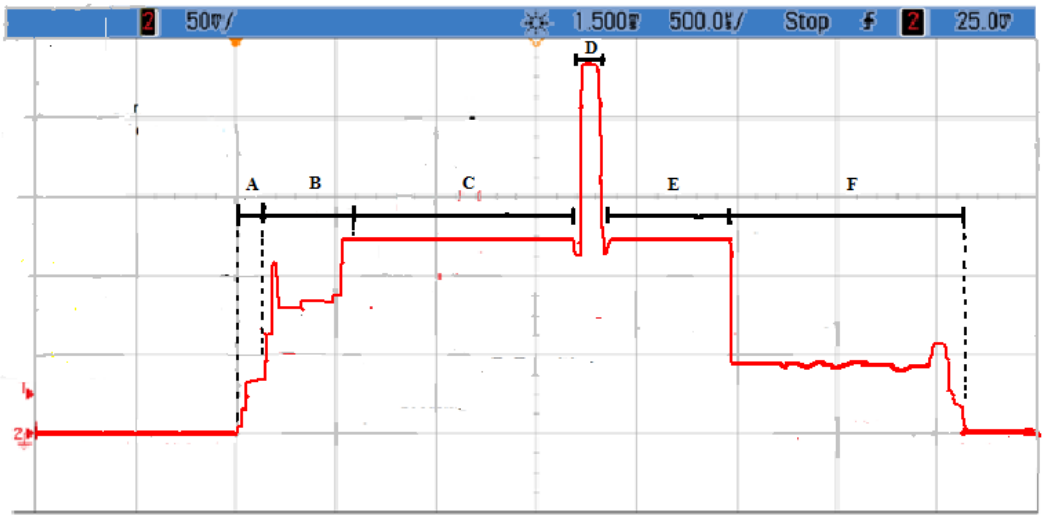


Fig. 4.1. Captura de la forma de onda durante un evento de conexión con los diferentes estados

Como se ve en la figura, se producen cambios tanto del consumo de corriente producido por los varios estados en los que pasa este evento de conexión como del tiempo de ejecución de cada uno de ellos.

Los estados por los que pasa son los siguientes:

- Estado A: Wake up del Waspmote

  El dispositivo Waspmote se despierta desde un nivel de dormido para proceder a realizar las acciones que se hayan ejecutado.

- Estado B: Proceso On del módulo XBee 802.15.4

  El módulo 802.15.4 se enciende. El pico que aparece ahí es a causa del propio encendido del mismo, estabilizándose al final en un estado constante.

- Estado C: Pre-procesamiento

  En este estado hay un tiempo de procesamiento previo al envío del paquete. Los XBees tienen que escuchar el canal al que se ha conectado y cuando comprueban que no hay nadie transmitiendo, se lanzan a enviar. Aparte, el microprocesador del XBee activa los pines necesarios para la transmisión y eso requiere tiempo de espera.

- Estado D: Envío del paquete

  El módulo procede a enviar el paquete al receptor. Normalmente sale un pico con una diferencia de consumo considerable aunque el tiempo es mínimo.

- Estado E: Post-procesamiento

  Se puede decir que ese tiempo de espera, después del envio del paquete corresponde al del ACK. Todos los XBees lo tienen. Es un paquete que el receptor envía al emisor como confirmación de que aquel ha recibido correctamente el paquete enviado.

- Estado F: Wake down del Waspmote

  Como se ve en la gráfica, el apagado del módulo se hace instantáneo, con lo cual el siguiente proceso respecto al anterior, es el proceso de echarse a dormir el Waspmote. Como se ve el tiempo de ejecución es mayor que cuando se despierta.

Para realizar las mediciones, cada estado debe ser divido en secciones aproximándolo a rectángulos de altura constante con la corriente y el tiempo de medido en cada uno.

La siguiente figura, muestra esta división en secciones para cada estado:

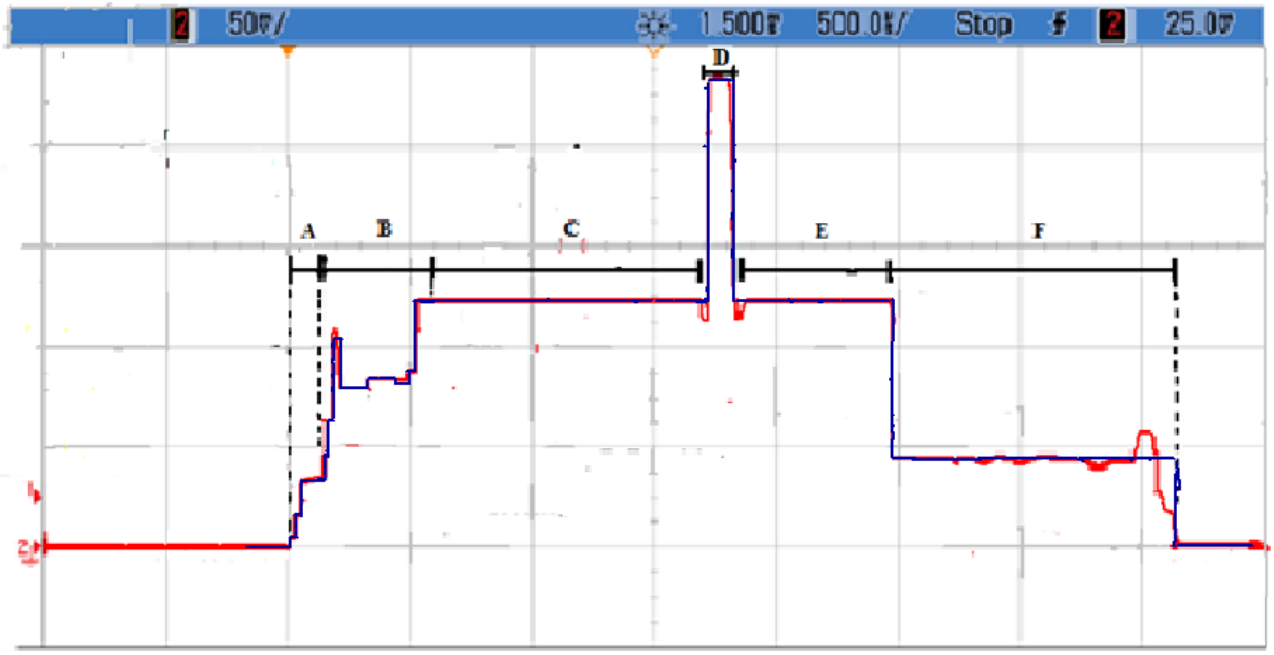


Fig. 4.2. Forma de onda de la corriente con los estados divididos en secciones

Utilizando los cursores del osciloscopio, se puede conseguir unas medidas exactas de la corriente y del tiempo para cada estado. Para algunos estado, como ocurre en el A, B y F, el consumo de la corriente no es contaste. Con lo cual para realizar estas mediciones tan precisas, la sección se puede dividir en secciones más pequeñas, como se ve en la figura.

Por lo tanto, usando las divisiones mostradas, se puede adivinar un valor actual promedio y realizar estimaciones bastante precisas de cada estado.

Una vez calculado en cada sección la corriente y el tiempo de ejecución, se realiza una tabla mostrando tales valores:

	Tiempo (ms)	**Corriente (mA)**
Sección 1	2	1.4
Sección 2	1	8.6
Sección 3	5	13
Sección 4	0.2	23.2
Sección 5	0.4	39.2
Sección 6	9.6	32.8
Sección 7	10.4	35.2
Sección 8	2	37.6
Sección 9	0.1	46.4
Sección 10	192.4	72

Sección 11	4.4	300
Sección 12	77.2	72
Sección 13	115	15.6

Tabla. 4.1. Medidas del tiempo y corriente en cada sección

Con los valores calculados, se calcula la carga energética total que consumiría todo el evento de conexión con la siguiente fórmula:

Carga total del evento de conexión en mA*mseg = [(Tiempo sección 1)*(Corriente sección 1) + (Tiempo sección 2)*(Corriente sección 2)+…] = 23382.72 mA*ms

Sabiendo esto y el tiempo total del proceso, podremos calcular la corriente promedio durante el evento de conexión:

Corriente promedio = [Carga total del evento de conexión (mA*ms)]/[Tiempo total del evento de conexión (ms)] = 23382.72 (mA*ms)/419.7 ms = 55.71 mA

Una vez calculada la corriente promedio, saber el total de horas de duración de la batería que conllevaría ejecutar todo el proceso global, es muy fácil. Sabiendo la capacidad de carga de la batería utilizada y aplicando la siguiente fórmula:

Carga batería utilizada = 6600 mA*h

Vida de la batería (h) = [Capacidad de la batería (en mA*h)]/[Corriente promedio del evento de conexión (mA)] = 118.47 h

Sabiendo las horas de aguante de la batería, podremos estimarlo en días o en años:

Vida de la batería (en días) = 118.47 h / 24 ≈ 5 días.

5. Medidas de ejemplos más característicos y cálculo del consumo y estimación de la vida de la batería de procesos globales

5.1. Introducción

Como hemos comentado anteriormente, las medidas se han hecho partiendo del Waspmote encendido continuamente, así sabremos cual es el consumo de cada proceso ejecutado. En ningún momento, la placa se lleva a *Sleep*. Sabiendo que se ha medido y caracterizado más de 200 medidas, vamos a incidir, en este apartado, en las más notables y características.

En cada una de las familias, se deben hacer múltiples mediciones, alrededor de 5 o 10 en cada caso.

Cabe destacar que cada consumo será sumado al que tienen los demás dispositivos que trabajan con él.

La siguiente tabla muestra los consumos del Waspmote en los 4 modos de operación posible, y así nos sirve de guía para observar y comprobar que, al medir el consumo en los diferentes estados con el osciloscopio, coincide con estos valores. Como se ha dicho, las mediciones de todos los dispositivos se ha hecho con el Waspmote en On, con lo cual se cogerá, de base, ese valor de consumo en ese estado.

	Consumption	Micro	Cycle	Accepted Interruptions
ON	15mA	ON	-	Synchronous and Asynchronous
Sleep	55µA	ON	32ms - min/hours/days	Synchronous (Watchdog) and Asynchronous
Deep Sleep	55µA	ON	1s – min/hours/days	Synchronous (RTC) and Asynchronous
Hibernate	0.06µA	OFF	1s – min/hours/days	Synchronous (RTC)

Fig. 5.1. Consumo de los 4 modos de funcionamiento del Waspmote

5.2. Medidas de las pruebas más características

A continuación, se va a exponer las pruebas más características de los tres grupos desarrollados (funcionalidades, sensores y módulos de comunicación), indicando el dispositivo medido, breve explicación de lo que se ha medido, el *link* que lleva al código ejecutado, las gráficas correspondientes como resultado del código ejecutado, una tabla del consumo del código ejecutado y una serie de comentarios acerca de la medida realizada (solo en los casos que sean necesarios).

El resto de las medidas están en el Anexo C del documento. Podréis acceder a los códigos y gráficas en el CD adjunto.

5.2.1. Funcionalidades del Waspmote

5.2.1.1. Proceso on y off de la tarjeta micro-SD.

En este ejemplo se va a medir el proceso y estado On y el proceso Off de la tarjeta micro-SD.

Se muestra las gráficas como resultado del ejemplo ejecutado:

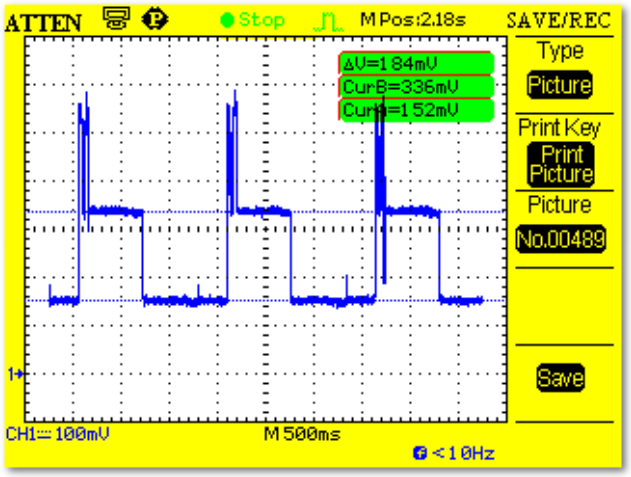


Fig. 5.2.a. Medida general cada 500 ms

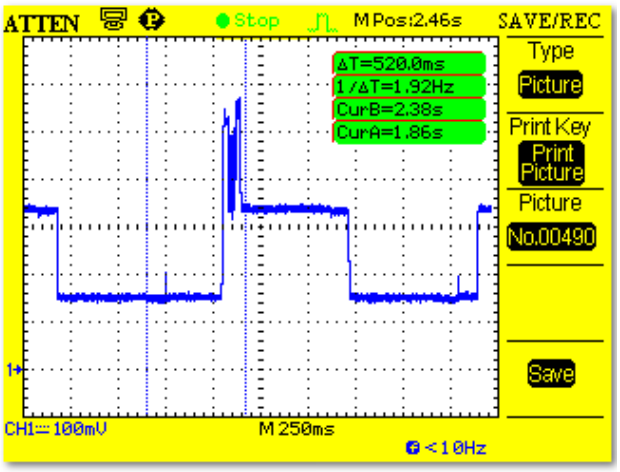


Fig. 5.2.b. Proceso a On

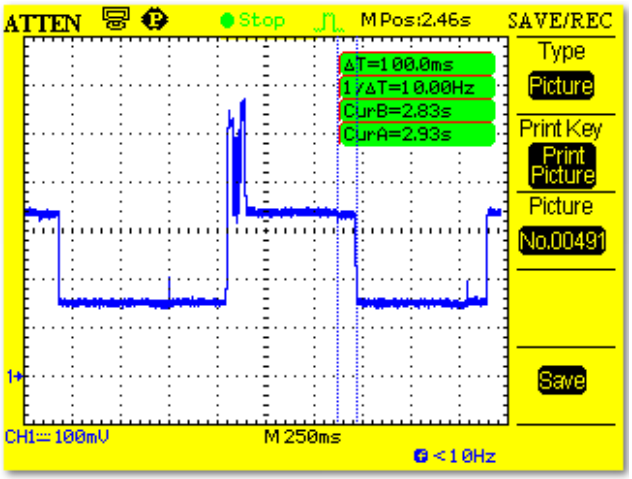


Fig. 5.2.c. Proceso a Off

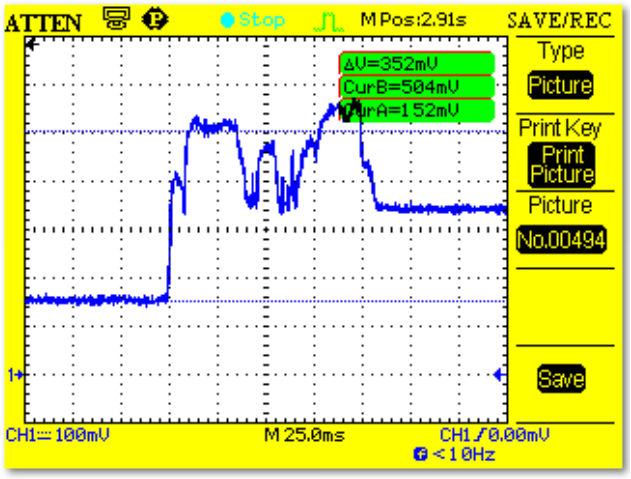


Fig. 5.2.d. Pico encendido de la tarjeta

Cabe comentar que, aunque sepamos, aplicando la función `millis`() (véase el ejemplo ejecutado), el tiempo de ejecución de cada acción, no sabemos en qué punto empieza y acaba cada una de ellas, por lo tanto, para distinguirlo, se añaden una serie de funciones antes y después de la acción que se ejecuta (se verán en el ejemplo comentadas, ya que solo se utiliza para tal fin, no para ser medidas). Estas funciones lo que hacen es poner el pin digital 8 en alto y bajo para, así forzar un consumo extra que pueda ser observado con facilidad en el osciloscopio y facilitar mejor la distinción entre dónde y dónde va a durar cada acción:

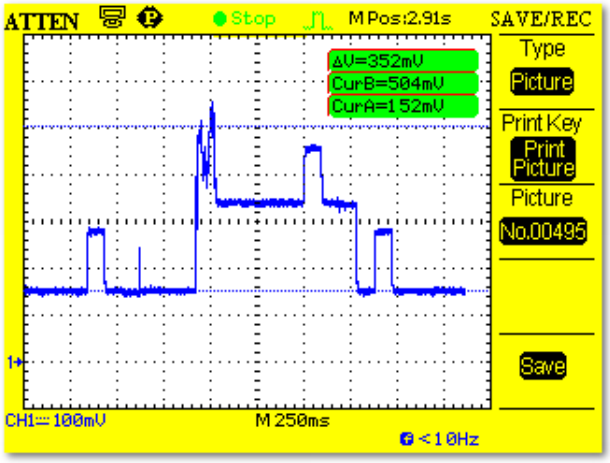


Fig. 5.2.e. Poniendo el pin digital en alto y bajo

En la siguiente tabla se muestra el consumo de los diferentes procesos, de acuerdo al código que se ha ejecutado:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	520	7.78	4044
Estado On	500	18.4	9200
Proceso Off	100	18	1800

Tabla 5.1. Consumo en los diferentes procesos del On/Off de la tarjeta SD

Se optó por poner un delay de 500 ms entre estado y estado para poder distinguir bien los procesosy que era suficiente para tal fin. Pero siguiendo el mismo proceso para todos, se toma como referencia tanto para el estado On como el Off de todos los estados "indefinidos" una duración de 1000 ms para que todos duren igual y sean exactos. Con lo cual, el consumo en el estado On sería:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Estado On	1000	18.4	18400

Tabla 5.2. Consumo en el estado On en 1000 ms

5.2.1.2. Modo Hibernate del Waspmote

Este ejemplo muestra cómo configurar Waspmote en el modo de consumo de energía más bajo (llevarlo a hibernate), desconectando toda la placa salvo el RTC, que se alimenta con la batería auxiliar.

Se muestra las gráficas como resultado del ejemplo ejecutado:

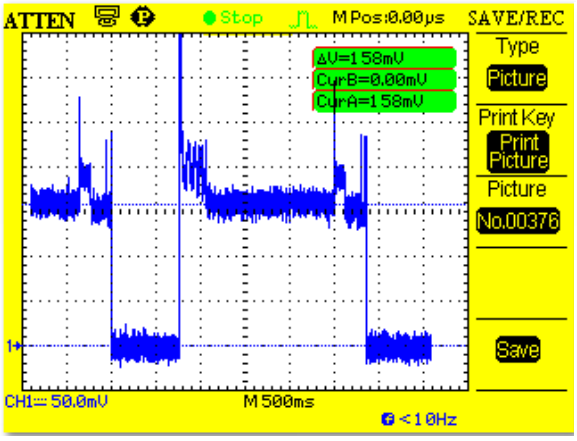


Fig. 5.3.a. Medida general del proceso

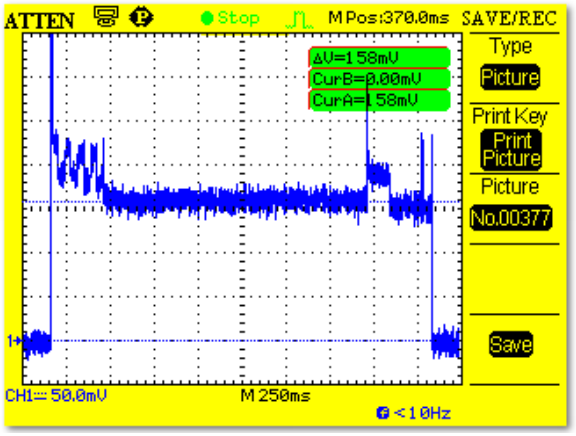


Fig. 5.3.b. Zoom de la medida general

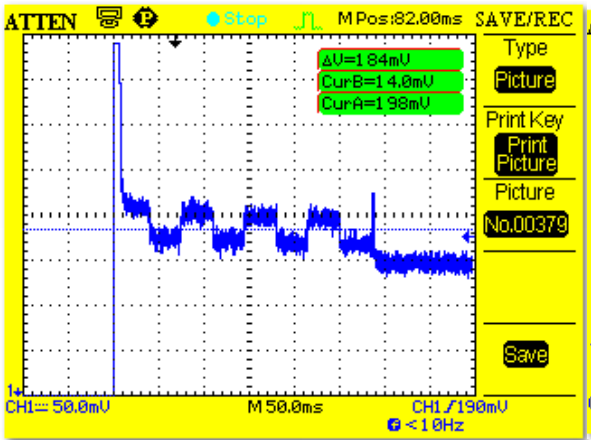


Fig. 5.3.c. Zoom de los picos del principio

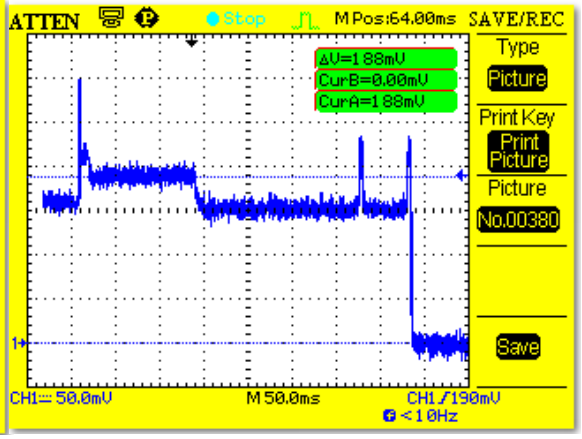


Fig. 5.3.d. Proceso de irse a dormir

Aplicando el mismo proceso que el ejemplo anterior, podemos averiguar hasta donde dura cada proceso que se produce en el modo *hibernate*. Se ve que, aunque se haya puesto un tiempo de estado en *hibernate* de 2 sg, no está en este estado todo ese tiempo, ya que se produce un tiempo de despierte desde *hibernate*, un tiempo de comprobación de si va a entrar a *hibernate* y posteriormente un tiempo de irse a dormir o irse a *hibernate*, *por lo tanto* el estado en *hibernate* se reduce respecto al tiempo programado.

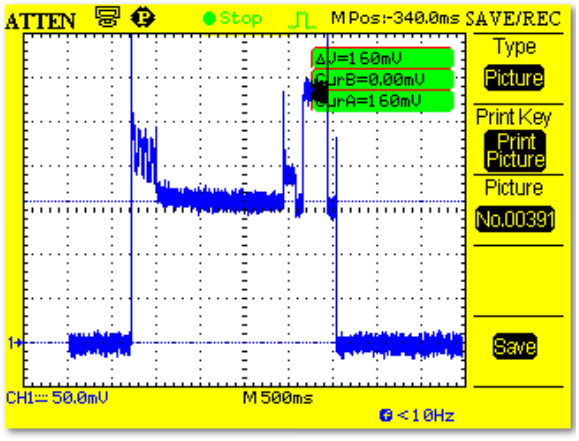


Fig. 5.3.e. Poniendo el pin digital en alto y bajo

En la siguiente tabla se muestra el consumo de los diferentes procesos:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Despertarse	1915	16.72	32022
Echarse a dormir	247	15.92	3933.2
Durmiendo	775	0.00006	0.0465

Tabla 5.3. Consumo en los diferentes procesos del modo hibérnate de Waspmote

5.2.2. *Placas de Sensores integrados*

5.2.2.1. Estado On y Off de la placa de Agricultura

En este ejemplo se va a medir el proceso y estado On y proceso Off de la placa de agricultura.

Se muestra las gráficas como resultado del ejemplo ejecutado:

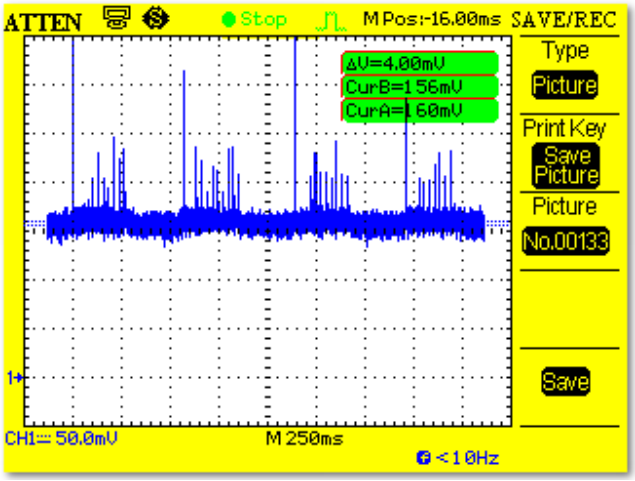


Fig. 5.4.a. Medida general cada 300 ms

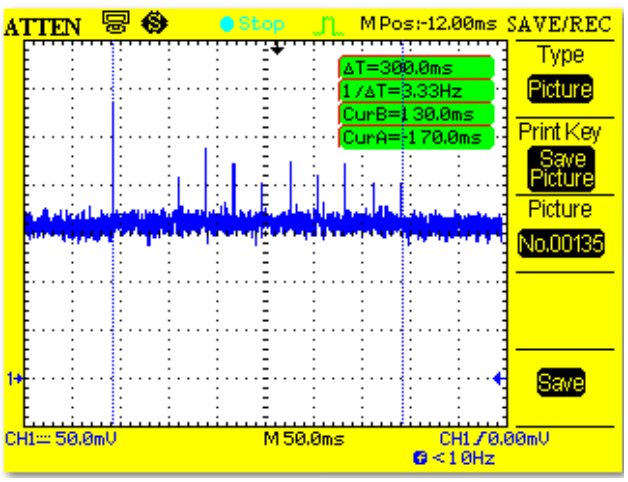


Fig. 5.4.b. Estado On de la placa

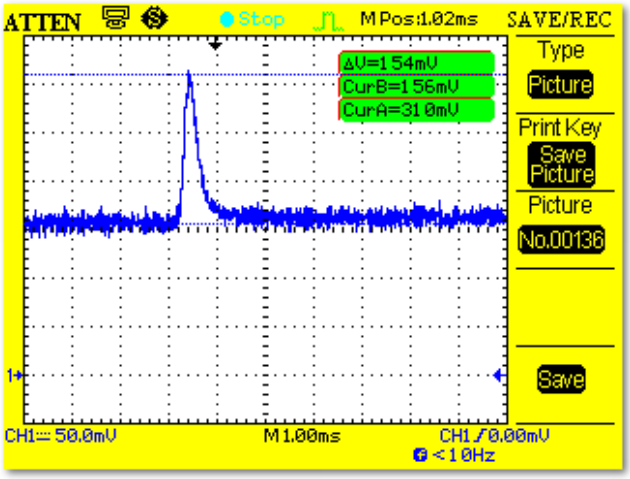


Fig. 5.4.c. Corriente del pico del On

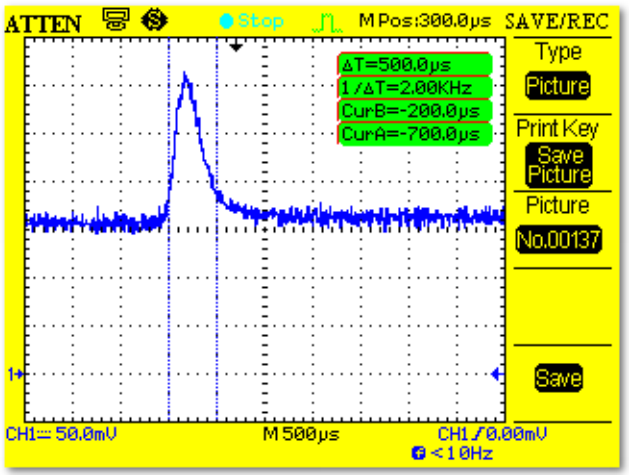


Fig. 5.4.d. Tiempo de ejecución del On

Los picos que aparecen en el estado On de la placa son debidos al conversor dc-dc. El gordo del principio es debido a la carga de condensadores, y luego aparecen picos, periódicos, con mayor frecuencia cuando mayor sea la corriente de salida del conversor.

En la siguiente tabla se muestra el consumo del proceso y estado On y proceso Off:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	0.5	10.78	5.39
Estado On	300	0.455	136.61
Proceso Off	0	0.455	0

Tabla 5.4. Consumo en los diferentes procesos del on/off de la placa Agricultura

Se optó por poner un delay de 300 ms entre estado y estado. Pero siguiendo el mismo proceso para todos, se toma como referencia tanto para el estado On como el Off de todos los estados “indefinidos” una duración de 1000 ms para que todos duren igual y sean exactos. Con lo cual, el consumo en el estado On sería:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Estado On	1000	0.46	460

Tabla 5.5. Consumo en el estado On en 1000 ms

5.2.2.2. Lectura Sensor de Presión de la placa de Gases

En este ejemplo se va a leer el valor del sensor de presión atmosférica cada segundo. Para ello, será necesario encender dicho sensor, previamente.

Se muestra las gráficas como resultado del ejemplo ejecutado:

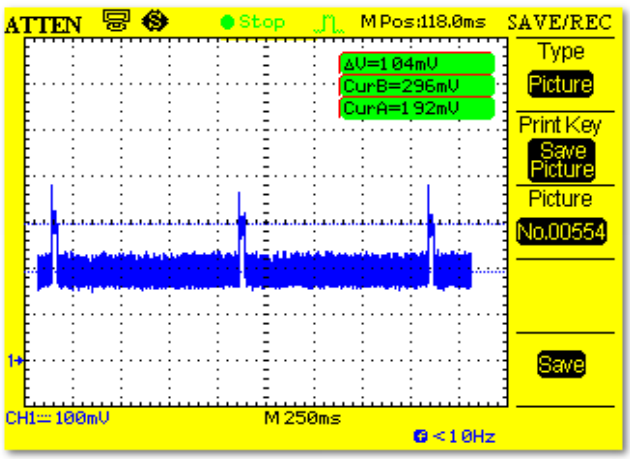


Fig. 5.5.a. Medida general cada segundo

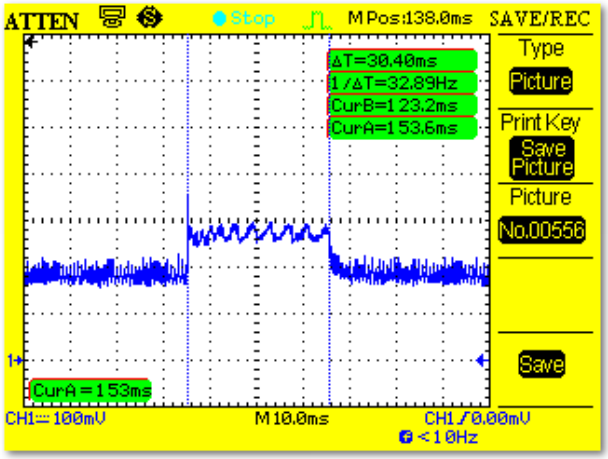


Fig. 5.5.b. Tiempo ejecución del proceso

Como se muestra en las gráficas, ejecutar las funciones de puesta en On, puesta a Off y lectura del sensor son instantáneas.

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso Total	30	10.4	312

Tabla 5.6. Consumo proceso total lectura sensor de presión en la placa de Gases

Comentarios:

- Si se observa el ejemplo ejecutado, el delay de 30 ms que se pone justo después de encender el sensor, es necesario ya que funciona como tiempo de respuesta del sensor. Este tiempo (de estabilización) es el que tarda el sensor en ofrecer una señal válida y precisa desde que se alimenta, debido a que a la señal de alimentación le lleva este tiempo estabilizarse. Si se observa en el datasheet correspondiente a este sensor, el tiempo de respuesta está en 20 ms. Aquí se puso 30 ms porque en las pruebas de calibración que se realizó previamente, se verificó que el sensor tarda algo más en alcanzar esta estabilización. Posiblemente, si se redujera, seguiría funcionando correctamente pero debemos evitar posibles errores.

- Se ha añadido en el setup, como se ve en el ejemplo, la función `pinMode (DIGITAL8, OUTPUT).` Es el pin que controla la alimentación del sensor de presión y necesario para que las mediciones salgan correctamente, ya que, en un principio, se midió con el código original (sin añadir esta función) y la gráfica que salía, no se correspondía con lo medido. Revisando la librería de Gases, se comprobó que no hacía esa acción, con lo cual, terminamos optando por añadirlo ahí y, así, cumplir con la finalidad que se buscaba.

5.2.2.3. Lectura del ADC de la placa de Prototipado.

Este ejemplo muestra como leer el conversor analógico-digital (ADC) de la placa de prototipado.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

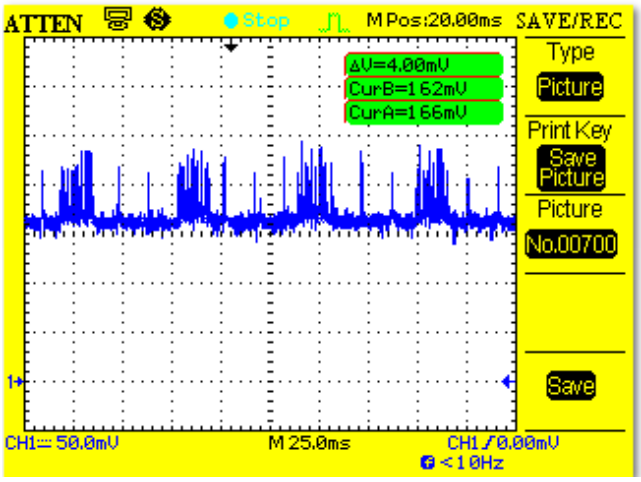


Fig. 5.6.a. Medida general del proceso

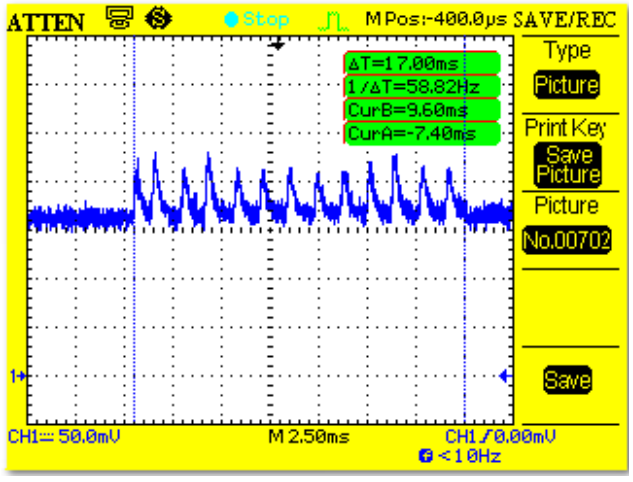


Fig. 5.6.b. Tiempo de conversión del ADC

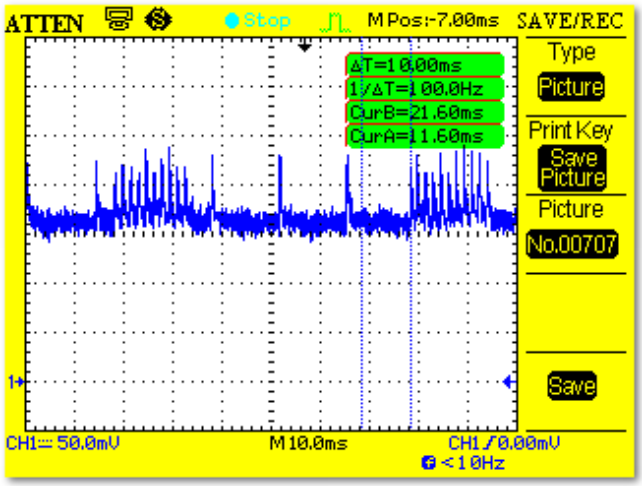


Fig. 5.6.c. Tiempo necesario para estabilización

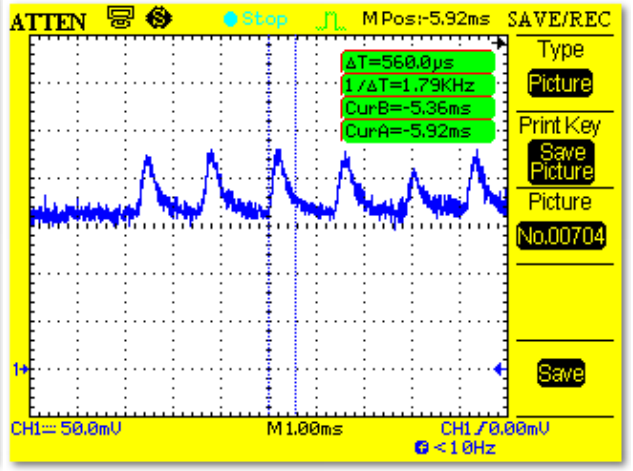


Fig. 5.6.d. Anchura pulsos durante la conversión

Como se ve en la imagen (c), es necesario dejar ese delay de 10 ms para alcanzar la estabilización de la señal y ofrecer una señal válida y precisa. Ejecutar la lectura del conversor es instantánea, pero luego le cuesta 17 ms convertir el valor.

El consumo de la lectura del ADC se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga Energética total (mA·ms)
Lectura ADC	27	1	27.18

Tabla C.85. Consumo lectura del ADC

5.2.2.4. Tomar fotos con la placa de Video Cámara

Este ejemplo consiste en configurar la placa de Video Cámara y hacer una foto. Para ello se necesita tener conectado previamente, el módulo 3G, ya que, solo funciona con él y se requiere de una tarjeta de memoria micro-SD, que se colocará en el módulo. En ella se habrá creado una carpeta llamada “Pictures” que el código seleccionará para guardar ahí las fotos que se hagan.

Se muestra las gráficas como resultado del ejemplo ejecutado:

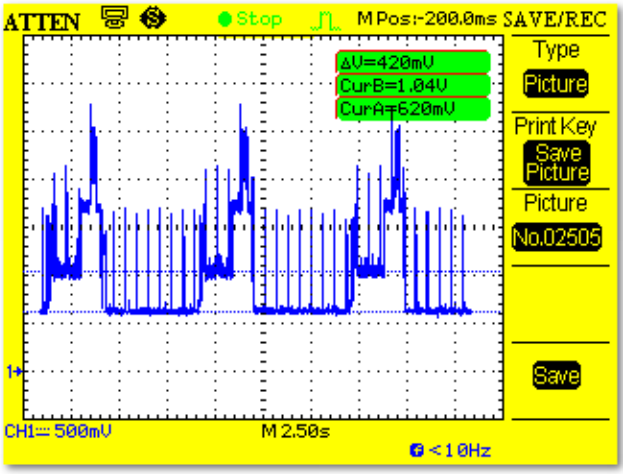


Fig. 5.7.a. Medida general cada 5 segundos

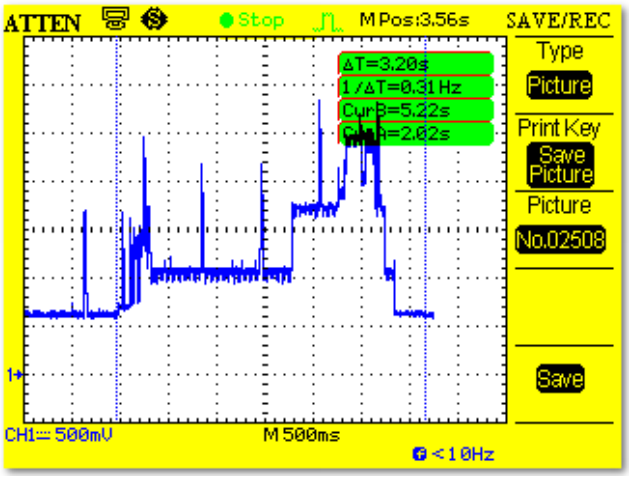


Fig. 5.7.b. Duración total proceso

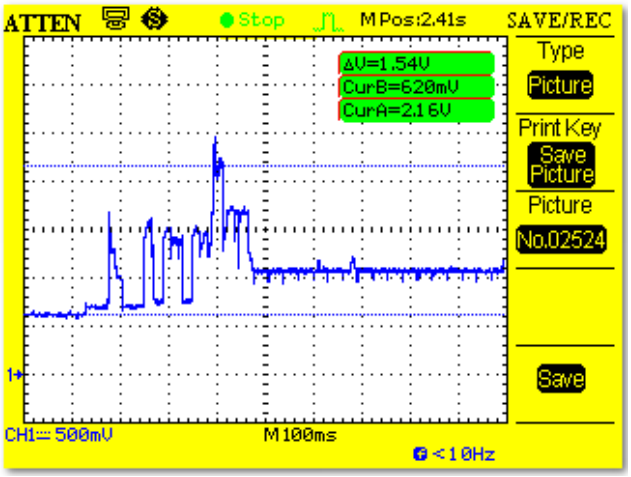


Fig. 5.7.c. Proceso On de la placa

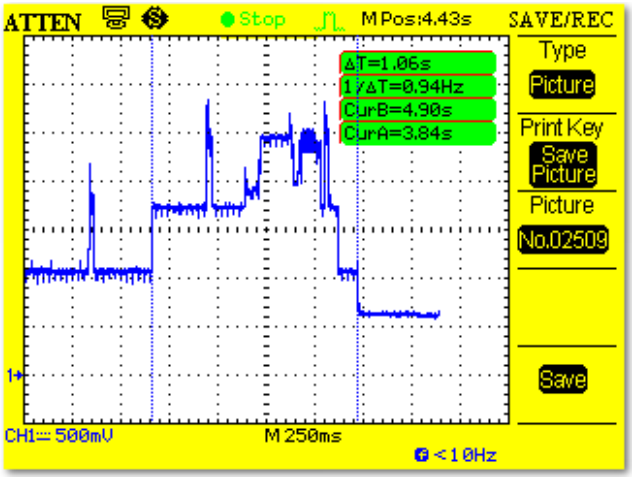


Fig. 5.7.d. Duración del proceso de hacer foto

En la siguiente tabla se muestra el consumo de los diferentes procesos, de acuerdo al código que se ha ejecutado. Partimos desde el estado On del módulo 3G, ya que, como hemos dicho antes, se requiere que este previamente encendido.

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Start Camara	540	53.83	29072
Configuración camara	1260	45.58	57432
Hacer foto	1060	112.77	119536
Stop Camara	320	2	640

Tabla 5.8. Consumo en los diferentes procesos del ejemplo “hacer foto” de la placa de videocamara

En la siguiente tabla se muestra el consumo de todo el proceso completo:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso Total	3200	64.58	206680

Tabla 5.9. Consumo del proceso total

5.2.3. Módulos de comunicaciones

5.2.3.1. Envío de paquetes por broadcast y en modo cifrado a través del módulo digiMesh

En este ejemplo, se muestra cómo enviar paquetes por Broadcast (a todos los nodos de la red) a través del módulo DigiMesh y en modo cifrado. La dirección de difusión 0x000000000000FFFF se especifica como dirección de destino. Con respecto a la carga útil de datos se ha optado por poner el máximo, 73 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

Aunque, para que los paquetes se envíen de forma cifrada, previamente se ha tenido que cargar un código para activar la función que lleva a tal fin.

Se muestra las gráficas como resultado del ejemplo ejecutado:

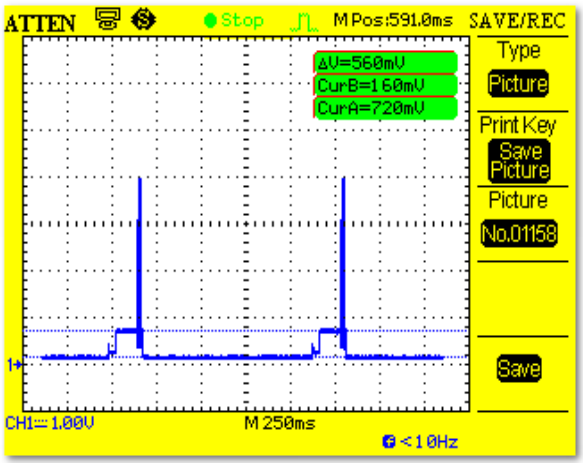
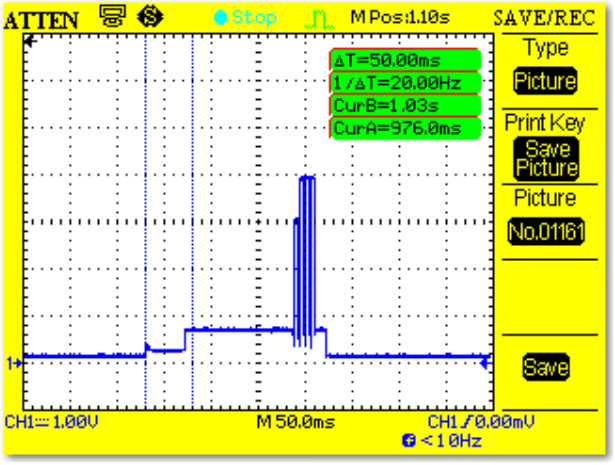


Fig. 5.8.a. Medida general cada segundo



5.8.b. Proceso On del módulo DM

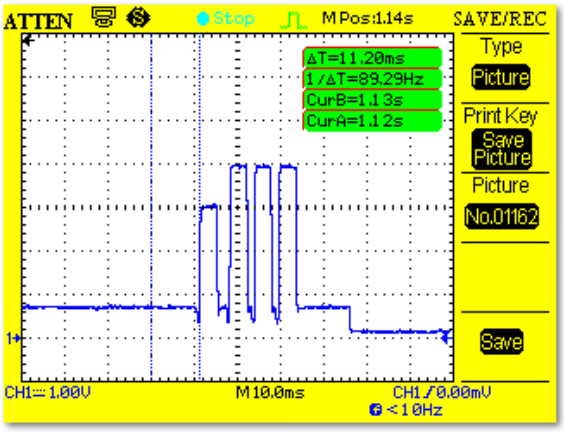


Fig. 5.8.c. Tiempo pre-proceso al envío de paquete

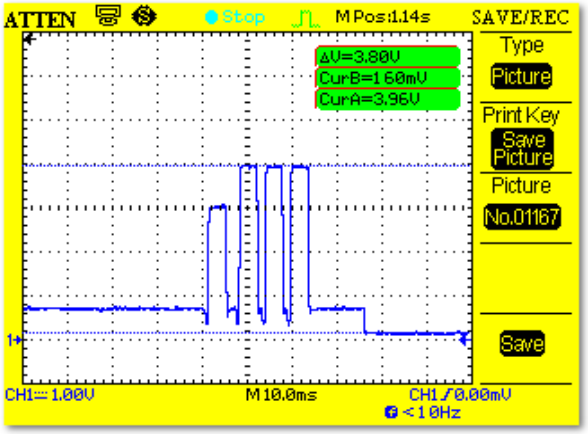


Fig. 5.8.d. Retransmisiones del paquete

Como se observa en las últimas gráficas, hay 4 picos de envíos de datos. Esto es debido a que le comando MT, que define el número de transmisiones de difusión adicionales, por defecto, es 3. Como todos los paquetes de difusión se transmiten MT+1 veces para asegurarse de que se recibe, entonces se producen 4 transmisiones, que son los 4 picos que se ven ahí.

Si vamos al código, el delay de 100 ms puesto después de encender el módulo, ha sido necesario añadirlo ya que sin él se producían errores de envío.

En la siguiente tabla se muestra el consumo de todo el proceso completo:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso Total	192	72.69	13957.12

Tabla 5.10. Consumo del proceso total de envió de paquetes por broadcast y cifrado a través de DM

5.2.3.2. Envío de archivos al servidor FTP desde Waspmote a través del módulo 3G

En este ejemplo se muestra como subir un archivo a un servidor FTP desde la tarjeta micro-SD de Waspmote. Para ello, dejamos previamente encendido el 3G y que esté conectado a la red. El archivo creado, llamado "/FILESD1" tiene un contenido de 100 bytes y la carpeta a la que se va a subir en el servidor FTP se ha llamado "/ftp_FILESD1".

Se muestra las gráficas como resultado del ejemplo ejecutado:

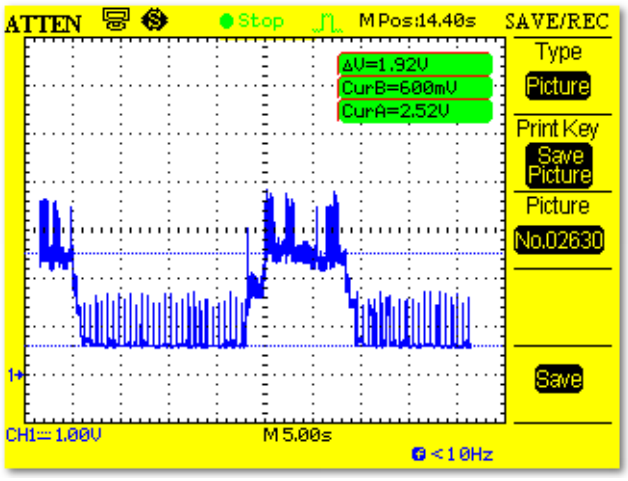


Fig. 5.9.a. Medida general cada 5 segundo

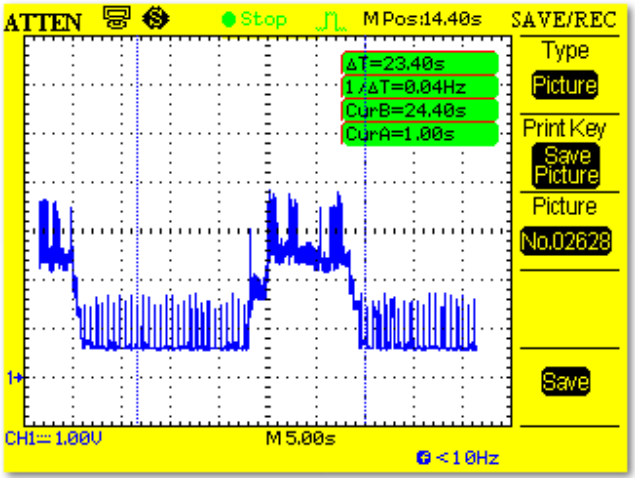


Fig. 5.9.b. Tiempo total del proceso completo

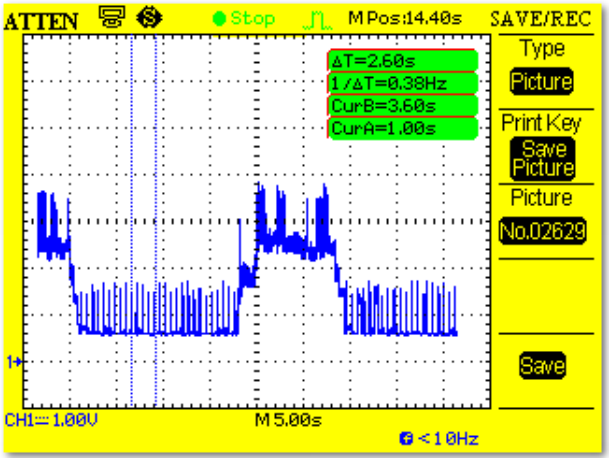


Fig. 5.9.c. Tiempo de ejecución de la configuración FTP

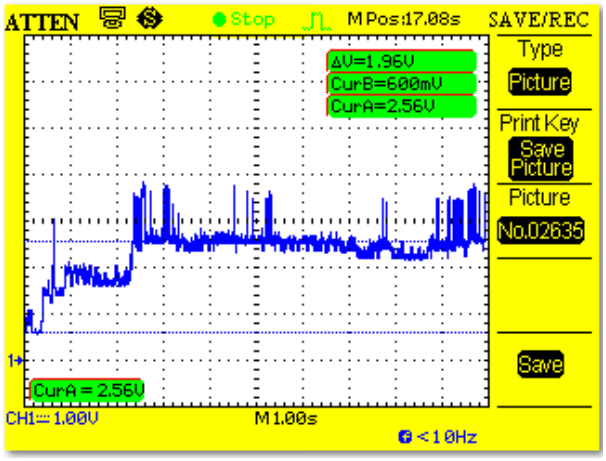


Fig. 5.9.d. Inicio del "subido del archivo"

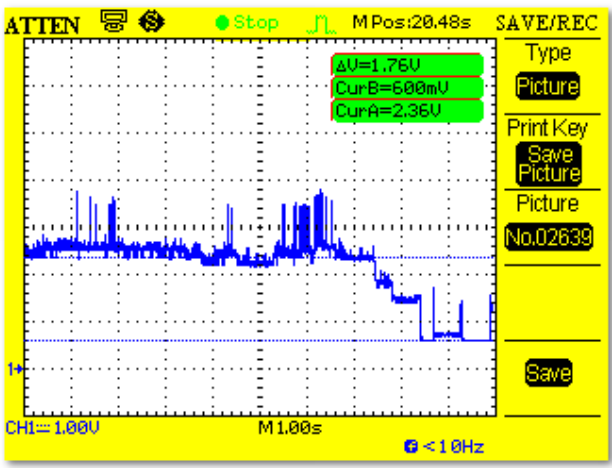


Fig. 5.9.e. Final del "subido del archivo"

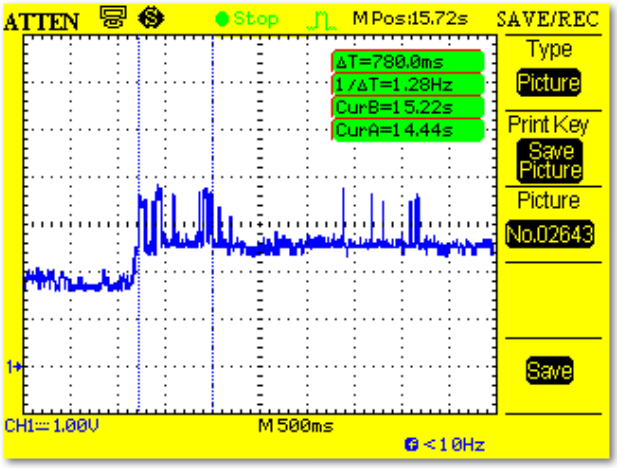


Fig. 5.9.f. Tiempos de los picos de inicio

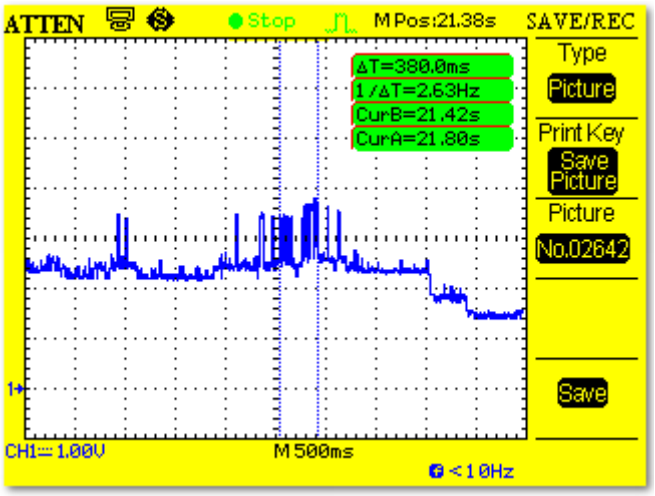


Fig. 5.9.g. Tiempo de los picos del final del proceso

En la siguiente tabla se muestra el consumo de todo el proceso completo:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso Total	23400	82.37	1927560

Tabla 5.11. Consumo proceso total del envio de archivos al servidor FTP desde el módulo 3G

Cabe destacar que al subir una vez el archivo con ese nombre al servidor FTP, cuando se vuelve a subir otra vez, da error. Esto es porque no se pueden subir más de una vez el mismo archivo con el mismo nombre. Se puede solucionar de dos formas:

- Borrando el archivo que se ha subido desde el servidor.
- Crear otro archivo con otro nombre en la tarjeta y subirla al servidor.

5.3. Cálculo del consumo en procesos globales.

Se ha calculado y caracterizado el consumo eléctrico de todos y cada uno de los distintos modos de funcionamiento de la plataforma Waspmote incluyendo sus sensores y módulos de comunicaciones asociados, a través de medidas experimentales para poder realizar estimaciones de autonomía de su batería de alimentación realizando lo que se conoce como "Energy Guide" presentando varias opciones para que el cliente o usuario y en base a las opciones selecionadas, pueda calcular los meses o años de la batería del dispositivo.

Aquí se expondrán una serie de procesos globales que le servirán al usuario de guía para el cálculo de los demás o de los que él quiera seleccionar.

Para el cálculo correcto de la vía de la batería, los procesos global incluyen desde que el Waspmote se despierta hasta que se echa a dormir, por eso ha sido importante calcular el consumo de los modos de operación del Waspmote.

Hay que tener en cuenta que cada consumo será sumado al que tienen los demás dispositivos que trabajan con él: filosofía modular. El consumo del Waspmote en el estado On es de 15.6 mA.

Los procesos globales que se han elegido para el cálculo de su consumo y así estimar la vida de la batería de acuerdo al seleccionado son los siguientes:

5.3.1. *Proceso de encendido del Waspmote + encendido del módulo DM + envío paquetes por Broadcast y modo cifrado + apagado del módulo + apagado del Waspmote*

Se ha calculado el consumo del ejemplo "envío de paquetes por Broadcast y modo cifrado a través de DM", englobando el proceso On del XBee, envío del paquete y proceso Off del XBee.

Teniendo en consideración las siguientes tablas:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Despertarse	8	9.55	76.4
Echarse a dormir	115	15.6	1794
Durmiendo	877	0.055	48.23

Tabla 5.12. Consumos de cada proceso del Deep Sleep deWaspmote

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso Total	192	72.69	13957.12

Tabla 5.13. Consumo proceso total envío paquetes a través de DM

Podemos calcular el consumo del proceso global:

$$Carga\ Energética\ total\ (mA * ms) = [9.55\ mA * 8\ ms] + [(72.69\ mA + 15.6\ mA) * 192\ ms] + [115\ ms * 15.6\ mA] = 18822.08\ mA * ms$$

$$Consumo\ promedio\ de\ corriente\ (mA) = \frac{Carga\ Energética\ total\ (mA * ms)}{Tiempo\ total\ empleado} = 59.75\ mA$$

Sabiendo que la batería que utilizamos tiene una capacidad de 6600 $mA * h$, podemos calcular la vida de ella:

$$Vida\ de\ la\ batería\ (h) = \frac{Capacidad\ de\ la\ batería\ (mA * h)}{Consumo\ promedio\ de\ corriente\ (mA)} = \frac{6600\ mA * h}{59.75\ mA} = 110.46\ h$$

Si la estimamos en días, son aprox. 5 días.

Aquí se muestra la tabla general del proceso:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso Total	315	59.75	18822.08

Tabla 5.14. Consumo proceso total

5.3.2. *Proceso de encendido del Waspmote + encendido placa Gases+ encendido sensor NO_2 + lectura del sensor + apagado sensor + apagado placa gases + apagado del Waspmote*

El tiempo de ejecución del proceso a On y Off es instantáneo.

Cabe destacar que, en esta placa, por motivo del HW diseñado en la misma, el estado OFF se queda en 16 mA, no bajan a los 15.6 mA que es lo normal en el estado On del Waspmote. Ese consumo extra de 0.4 mA que se produce hay que tenerlos en cuenta.

Teniendo en consideración las siguientes tablas:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Despertarse	8	9.55	76.4
Echarse a dormir	115	15.6	1794
Durmiendo	877	0.055	48.23

Tabla 5.14. Consumos de cada proceso del Deep Sleep deWaspmote

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso On	0	24.4	0
Estado On	1000	4	4000
Proceso Off	0	4	0

Tabla 5.15. Consumo total de cada proceso On/Off placa Gases

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso Total	40000	55.2	2208000

Tabla 5.16. Consumo total de la lectura del Sensor NO_2

Podemos calcular el consumo del proceso global:

$Carga\ Energética\ total\ (mA*ms) = [9.55\ mA*8\ ms] + [(55.2\ mA + 16\ mA + 4\ mA) * 40000\ ms + 115\ ms * 15.6\ mA = 3009870.4\ mA*ms$

$$Consumo\ promedio\ de\ corriente\ (mA) = \frac{Carga\ Energética\ total\ (mA*ms)}{Tiempo\ total\ empleado} = 75.01\ mA$$

Sabiendo que la batería que utilizamos tiene una capacidad de 6600 $mA*h$, podemos calcular la vida de ella:

$$Vida\ de\ la\ batería\ (h) = \frac{Capacidad\ de\ la\ batería\ (mA*h)}{Consumo\ promedio\ de\ corriente\ (mA)} = \frac{6600\ mA*h}{75.01\ mA} = 88\ h$$

Si la estimamos en días, son aprox. 4 días.

Aquí se muestra la tabla general del proceso:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso Total	315	59.75	18822.08

Tabla 5.17. Consumo proceso total

5.3.3.- Proceso de encendido del Waspmote+encendido módulo WiFi+ conexión punto de acceso+TCP client+ apagado del módulo+ apagado del Waspmote

Se ha calculado el consumo del ejemplo “TCP client”, englobando el proceso On del módulo Wifi, conexión a un punto de acceso, creación de una conexión TCP, envío de tramas a la conexión TCP y por último, proceso Off del XBee.

Teniendo en consideración las siguientes tablas:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Despertarse	8	9.55	76.4
Echarse a dormir	115	15.6	1794
Durmiendo	877	0.055	48.23

Tabla 5.18. Consumos de cada proceso del Deep Sleep deWaspmote

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso Total	9520	31.8434	303149.2

Tabla 5.19. Consumo proceso total ejemplo "TCP client"

Podemos calcular el consumo del proceso global:

$Carga\ Energética\ total\ (mA*ms) = [9.55\ mA*8\ ms] + [(31.84\ mA + 15.6\ mA)*$
$9520\ ms+115\ ms*15.6\ mA=453499.2\ mA*ms$

$$Consumo\ promedio\ de\ corriente\ (mA) = \frac{Carga\ Energética\ total\ (mA*ms)}{Tiempo\ total\ empleado} = 47.02\ mA$$

Sabiendo que la batería que utilizamos tiene una capacidad de 6600 $mA*h$, podemos calcular la vida de ella:

$$Vida\ de\ la\ batería\ (h) = \frac{Capacidad\ de\ la\ batería\ (mA*h)}{Consumo\ promedio\ de\ corriente\ (mA)} = \frac{6600\ mA*h}{47.02\ mA}$$
$$= 140.36\ h$$

Si la estimamos en días, son aprox. 6 días.

Aquí se muestra la tabla general del proceso:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso Total	9643	47.02	453499.2

Tabla 5.20. Consumo proceso total

6. Resultados.

A continuación, se muestra una tabla con el consumo de cada una de las acciones medidas de cada uno de los tres grupos analizados:

Columna1	Tiempo total (ms)	Corriente total (mA)	Carga Energética Total (mA·ms)
Proceso on Led 0	0	3,6	0
Estado on Led 0	1000	3,6	3600
Proceso off Led 0	0	3,6	0
Proceso on Led 1	0	3,2	0
Estado on Led 1	1000	3,2	3200
Proceso off Led 1	0	3,2	0
Lectura EEPROM	0	0	0
Escritura EEPROM	3,7	7	25,9
Lectura/escritura EEPROM	3,7	7	25,9
Proceso on RTC	2,2	1,3	2,86
Estado on RTC	1000	0,4	400
Proceso off RTC	0	0,4	0
Fijar fecha/hora RTC	1	0,8	0,8
Leer fecha/hora RTC	2	1,2	2,4
Leer temperatura RTC	2	1,2	2,4
Despertarse del Deep-Sleep	8	9,55	76,4
Echarse a dormir Deep-Sleep	115	15,6	1794
Duermiendo Deep-Sleep	877	0,055	48,23
Nivel de batería	1	0	0
Despertarse Hibernate	1915	16,72	32022
Echarse a dormir Hibernate	247	15,92	3933,2
Modo Hibernate	775	0,00006	0,0465
Proceso on ACC	2	1,1	2,2
Estado on ACC	1000	0,4	400

Proceso off ACC	1	0,8	0,4
Lectura ACC	3,32	0,96	3,18
Proceso on SD	520	7,78	4044
Estado on SD	1000	18,4	18400
Proceso off SD	100	18	1800
Escritura SD	9,6	27,77	266,68
Lectura SD	4,84	6,47	31,32
Proceso on placa Agricultra	0,5	10,78	5,39
Estado on placa Agricultura	1000	0,46	460
Proceso off placa Agricultura	0	0,46	0
Lectura presión Agricultura	100	11	1100
Lectura humedad Agricultura	100	3,22	322,25
Lectura LDR Agricultura	100	0,34	34
Lectura Sensirion Agricultura	396	0,424	168
Lectura Watermark Agricultura	340	3,44	1172
Lectura Anemómetro Agricultura	3240	0,6	1944
Lectura Veleta Agricultura	305	0,6	183
Interrupción Pluviómetro Agricultura	8,3	1,37	11,38
Proceso on Agricultura PRO	1	24,8	24,8
Estado on Agricultura PRO	1000	4,6	4600
Proceso off Agricultura PRO	0	4,6	0
Lectura Pt1000 Agricultura PRO	624	2	1248
Lectura dendrómetro Agricultura PRO	624	0,2	124,8
Lectura sensor SQ-110 y SU-100	250	0	0
Proceso on placa Eventos	22,4	1,76	39,42
Estado on placa Eventos	1000	0,3	300
Proceso off placa Eventos	0	0,3	0
Lectura Flexiforce PS-02 Eventos	28,4	1,02	29,018
Lectura FLX-C1-H	69	1,68	116,312
Lectura vibración Eventos	10	0,2	2
Lectura Temperatura Eventos	100	0,005	0,5

Lectura PTFA3415 Eventos	29	1,02	29,85
Lectura PFTA1103 Eventos	27,6	1	27,65
Lectura PTFA0100 Eventos	28,4	1,01	28,91
Lectura LDR Eventos	10	0,368	3,68
Lectura PIR Eventos	2360	3	7080
Lectura Efecto Hall Eventos	26	0,96	25,13
Lectura detección de líquido Eventos	10	0,005	0,05
Lectura Elongación Eventos	10	0,28	2,8
Lectura Humedad Eventos	10	0,568	5,68
Lectura de detección de líquido Eventos	10	0,005	0,05
Lectura flujo líquido FS100A y FS200B	1040	2,168	2254,72
Proceso on placa Gases	8	24,8	198,4
Estado on placa Gases	1000	3,7	3769,6
Proceso off placa Gases	0	3,7	0
Lectura humedad Gases	15000	0,9	13500
Lectura Temperatura Gases	100	0,005	0,5
Lectura CO Gases	993	5,23	5232
Lectura CO2 Gases	40000	82,8	3312000
Lectura O2 Gases	10	0	0
Lectura NO2 Gases	40000	55,9	2236000
Lectura NH3 Gases	313,2	16,7	5232
Lectura CH4 Gases	40000	104,7	41188000
Lectura TGS2610 Gases	40000	106,7	4268000
Lectura TGS2600 Gases	40000	81,7	3268000
Lectura TGS2602 Gases	40000	106,7	4268000
Lectura TGS2620 Gases	40000	77,5	3100000
Lectura MiCS-2610 Gases	40000	70,3	2812000
Lectura MiCS-5521 Gases	40000	58,3	2332000
Proceso on placa parking	6	114,208	685,25
Estado on placa parking	1000	24,8	24800
Proceso off placa parking	0	24,8	0

Lectura MFS parking	368	6,8	2502,4
Proceso on placa Smart Metering	1,12	12,6	14,11
Estado on placa Smart Metering	1000	0,26	260
Proceso off placa Smart Metering	0	0,26	0
Lectura corriente Smart Metering	50	0,2	10
Lectura célula de carga a 5V Metering	50	5,05	252,88
Lectura célula de carga a 10V Metering	50	36,26	1813,12
Lectura Flujo de agua a 3,3V Metering	1100	2,4	2640
Lectura Flujo de agua a 5V Metering	1100	5,6	6160
Lectura humedad Metering	15000	1,8	27000
Lectura Temperatura Metering	50	0,005	0,25
Lectura LDR Metering	50	0,6	30
Lectura lámina de desplazamiento a 3,3V	52	0,6	31,2
Lectura lámina de desplazamiento a 5V	50	2	100
Lectura ultrasonido WRA1 a 5V	1000	9,52	9522,04
Lectura ultrasonido EZ0 a 5V	1000	4,815	4815,2
Proceso on placa de Prototipado	5,72	82	496,05
Estado on placa de Prototipado	1000	0,82	822,6
Estado off placa de Prototipado	0	0,82	0
Lectura ADC Prototipado	27	1	27,18
Proceso on placa Smart Cities	0,44	33,12	14,6
Estado on placa Smart Cities	1000	0,45	450,07
Proceso off placa Smart Cities	0	0,45	0
Lectura temperatura Smart Cities	10	23,15	231,5
Lectura humedad Smart Cities	15000	2,01	30282
Lectura LDR Smart Cities	10	19,95	199,5
Lectura Partículas Smart Cities	2000	22,45	44898,88
Lectura Detección de Grietas Smart Cities	10	18,48	184,892
Lectura Ruido Smart Cities	3200	1,2	3848
Lectura Ultrasonido WRA1 a 3,3V	2000	2,99	5992,64
Lectura Ultrasonido EZO a 3,3V	2000	2,4	4808,96

Proceso on Placa de Radiación	2000	28,14	56291,25
Estado on Placa de Radiación	1000	22	22000
Proceso off Placa de Radiación	0	22	2
Lectura Radiación	5000	4,8	24000
Proceso on placa Smart Water	1	300,61	300,61
Estado on placa Smart Water	1000	4,03	4032
Proceso off placa Smart Water	0	4,03	0
Lectura Pt1000 Smart Water	1260	3,68	4636,8
Lectura Conductividad Smart Water	1360	2,37	3223,2
Lectura Oxígeno Disuelto Smart Water	1360	0,825	1122
Lectura pH Smart Water	2640	0,92	2428,8
Lectura ORP Smart Water	1360	0,932	1267,52
Lectura DI Smart Water	1360	0,97	1319,2
Proceso on placa de Video Cámara	540	53,83	29072
Estado on placa de Video Cámara	1000	44,256	44256
Proceso off placa de Video Cámara	320	2	640
Proceso on Leds IR Video Cámara	0	384	0
Estado on Leds IR Video Cámara	1000	384	384000
Proceso off Leds IR Video Cámara	0	384	0
Hacer foto Video Cámara	3200	64,58	206680
Grabar Video	43900	116,9	5132240
Interrupción PIR, hacer foto y subirla al FTP	36000	205,418	7395072
Proceso on módulo 802.15.4	50	39,24	1962,28
Estado on módulo 802.15.4	1000	57,2	57200
Proceso off móudlo 802.15.4	0	57,2	0
Envío paquetes unicast y no cifrado 802	298	56,81	16931,56
Envío paquetes broadcast y no cifrado 802	296	56,76	16801.16
Envío paquetes unicast y cifrado 802	308	56,8	17495.56
Envío paquetes broadcast y cifrado 802	306	56,748	17365,16
Lectura comandos AT 802	207	53,87	14546,28
Obtención RSSI 802.15.4	760	0	0

Proceso on módulo 868	54,4	55,1	2997,448
Estado on módulo 868	1000	60,4	60400
Proceso off módulo 868	0	60,4	0
Envío paquetes unicast y no cifrado 868	137	182,93	25062,648
Envío unicast y no cifrado con reintentos 868	264	237,12	62601,44
Envío paquetes broadcast y no cifrado 868	282	266,66	75200,64
Envío paquetes unicast y cifrado 868	140	186,28	26079,84
Envío paquetes broadcast y cifrado 868	286	288,51	82514,24
Lectura comando AT 868	186	60,27	11211,84
Obtención RSSI 802.15.4 868	5040	1,283	6468
Proceso on módulo DM	50	17,24	862,24
Estado on módulo DM	1000	55,6	55600
Proceso off módulo DM	0	55,6	0
Envío paquetes unicast y no cifrado DM	1324	59,94	75393,28
Envío paquetes broadcast y no cifrado DM	102	77,86	7942,24
Envío paquetes unicast y cifrado DM	1340	57,47	77022,4
Envío paquetes broadcast y cifrado DM	192	72,69	13957,12
Lectura comandos AT DM	240	47,6	11426,24
Obtención RSSI	4720	0,693	3272,64
Proceso on módulo ZigBee	470	21,88	10284,52
Estado on módulo ZigBee	1000	44,4	44400
Proceso off módulo ZigBee	0	44,4	0
Conexión router al coordinador ZB	642	27,91	17921,32
Envío paquetes unicast y no cifrado ZB	3796	41,65	158131,72
Envío paquetes broadcast y no cifrado ZB	4680	42,42	198558,9
Envío paquetes unicast y cifrado ZB	3790	41,83	158572,52
Envío unicast y cifrado 2 retransmisiones ZB	3870	42,09	162906,52
Envío paquetes broadcast y cifrado ZB	4710	42,7	201134,52
Configuración modo cifrado Router ZB	6590	42,8	282086,12
Configuración modo cifrado Coordinador ZB	13990	43,7	611496,52
Proceso on Bluetooth	5800	15,03	87221,6

Estado on Bluetooth	1000	2,8	2800
Proceso off Bluetooth	200	2	400
Escaneo normal Bluetooth	6480	40,8	264384
Escaneo limitado Bluetooth	3480	40,8	141984
Escaneo con nombre descriptivo Bluetooth	24200	37,67	911760
Escaneo con nombre específico Bluetooth	6040	39,15	236468
Creación conexión transparente Bluetooth	16540	33,32	551232
Proceso on módulo BLE	80	2,33	186,4
Estado on módulo BLE	1000	8,4	8400
Proceso off módulo BLE	0	8,4	0
Escaneo normal BLE	5260	9,03	47520
Escaneo con nombre del dispositivo BLE	5920	12,32	72986,2
Escaneo limitado BLE	990	11,21	11102,84
Escaneo dispositivo BLE	206	20,33	4189,6
Configurar una conexión BLE	1198	5,5	6598,8
Configurar un escaneo BLE	5020	12,19	61202,4
Configurar avisos BLE	21000	0,62	1302
Conexión a un dispositivo B LE como maestro	888	7,34	6522,88
Proceso on módulo GPS	1570	36,48	57282,4
Estado on módulo GPS	10000	54,92	54925,4
Proceso off módulo GPS	0	54,92	0
Lectura datos GPS	65280	0,034	2277,6
Proceso on módulo Wi-Fi	2360	25,91	61147,64
Estado on módulo Wi-Fi	1000	32,4	32400
Proceso off módulo Wi-Fi	460	2	920
Unión punto acceso Wi-Fi	7140	29,67	211833,16
Envío mensajes de solicitud HTTP Wi-Fi	14660	33,72	494443,16
Envío mensajes de solicitud HTTP con tramas	17360	32,86	570483,16
Conexión clientes TCP	5700	0	0
Conexión clientes UDP	19660	33,74	663483,17
Proceso on módulo GPRS	8040	43,48	349589,88

Estado on módulo GPRS	1000	24,4	24400
Proceso off módulo GPRS	2880	2,8	6064
Conexión a la red GPRS	11320	33,28	365413,88
Envío SMS GPRS	4400	24,43	107520
Lectura URL con petición GET GPRS	24040	23,55	566144
Subida de archivos al servidor FTP GPRS	41000	91,17	3738240
Conexión TCP GPRS	32120	69,93	2246449,88
Conexión UDP GPRS	30520	57,47	1753989,88
Proceso on módulo 3G	10000	99,92	999237,56
Estado on módulo 3G	1000	82,4	82400
Proceso off módulo 3G	2100	2,4	5040
Conexión a la red 3G	22500	101,8	2290516,69
Envío SMS 3G	4520	36,86	166608
Lectura URL con petición GET 3G	26004	106,47	2768880
Envío archivo desde el Waspote al FTP 3G	23400	82,37	1927560
Envío archivo desde el módulo 3G al FTP	17600	154,44	2718300
Envío de emails con SMTP 3G	7880	147,96	116600
Envío datos a través de TCP 3G	15500	125,5	1945400
Envio de datos a través de UDP 3G	12800	114,9	1470840
Proceso on modo GPS 3G	320	57,56	18420
Estado on modo GPS 3G	1000	57	57000
Proceso off modo GPS 3G	320	2	640
Modo Stand- Alone GPS 3G	84480	5,66	478720
Modo MS-based GPS 3G	7480	4,915	36768,8

Tabla 6.1. Consumos de cada acción medida de los tres grupos analizados

Como se ve en la tabla de consumos, en total se midió 233 acciones que puede realizar la plataforma Waspmote utilizando las funcionalidades, los sensores y los módulos de comunicaciones que lo integran.

En total, se adaptaron 206 códigos para tal fin ejecutándose unas 1030 funciones.

Con respecto a las medidas de consumo, se realizaron más de 1000 capturas de pantalla de señales de consumo, ya que era necesario, según el resultado de las representaciones, incidir más en unas que en otras, facilitando el cálculo del consumo con mayor exactitud.

Observando la tabla de todas las pruebas realizadas, es decir de todas las acciones de las

funcionalidades del Waspmote, placas de sensores con los sensores que integran cada una y todos los módulos de comunicaciones, se comprobó lo que se pensaba en un principio de que los módulos de comunicaciones, en medidas generales, son los que más consumen sin duda, siguiendo en menor medida los sensores y por último las funcionalidades del Waspmote.

Respecto a los módulos de comunicaciones, las acciones que realiza los módulos 3G y GPRS se ve, con gran diferencia, un gran valor de consumo con respecto al resto de módulos.

Cabe destacar, que, aunque los proceso totales medidos de los módulos XBees, no consuman mucho, el envío, en sí, de los paquetes de datos producen un pico de consumo considerable de órdenes de 400 mA, consumiendo mas en modo broadcast que en unicast y en modo cifrado que sin cifrar.

Con respecto a los sensores, el que más consume con gran diferencia respecto a otros es la placa de Video Cámara ya que requiere la utilización del módulo 3G, siguiente con los sensores de la placa de Gases.

Cabe destacar, que las tareas que me llevaron menor tiempo de ejecución, fueron las funcionalidades del Waspmote, por sus funciones sencillas de ejecución y su fácil caracterización de su consumo, debido a sus sencillas gráficas gracias a una electrónica más sencilla.

Las placas de sensores me llevo mucho más tiempo debido a la gran variedad que tiene la empresa Libelium y que en la mayoría integra más de 10 sensores. Caracterizar cada consumo era un proceso lento y minucioso, ya que dependiendo del resultado de las representaciones de las señales eléctricas, requería un análisis más profundo, además de saber distinguir y diferenciar en ellas, cada proceso de ejecución, aunque el código utilizado, al llevar el mismo criterio para todos, era más fácil de llevar. En total se caracterizaron más de 70 sensores.

Los módulos de comunicaciones son los que me llevaron mucho más tiempo, sobre todo al principio a la hora de documentarse en cada uno de ellos. Hay que contar que tienen una electrónica más complicada. No se quedan atrás los códigos utilizados para caracterizar cada acción medida, ya que debido a esto, las señales son más complicadas de analizar que, como con los sensores, requieren un análisis mucho más profundo y diferenciar y distinguir cada proceso es mucho más complicado.

Los dispositivos que me llevaron más trabajo respecto a otros fueron:

- Placas de protopipado, agricultura, Smart Metering, gases, Smart Cities y la de Smart Water: debido a la aparición de múltiples picos en sus medidas por el conversor dc-dc (explicado en el apartado 5.2.2.1).
- Sensor de Presión Atmosférica de la placa de Gases (explicado en el apartado 5.2.2.2): Estudiar las posibles causas y luego llegar a la solución requirió de tiempo para corregir ese *bug*.
- Lectura del conversor ADC de la placa de prototipado: debido a su difícil interpretación de la señal resultante.
- Módulo de comunicación 802.15.14: saber interpretar el resultado de la señal resultante. Me llevo tiempo investigar (mirando en el datasheet del módulo) que comandos influen para que haya un tiempo de pre-procesado y post-`procesado en el envío de paquetes. (explicado en el apartado C.3 de los anexos).
- Módulo GPRS: debido a que el conector de la antena no hacía buen contacto con el módulo,

y la señal no llegaba por lo tanto no se producía la conexión. También dependiendo del momento. Había días que la señal era buena y llegaba y otras no. Por lo tanto, eso retrasaba la realización de mis mediciones.

- Módulo Wifi: ese es el módulo que más complicaciones me ha dado a la hora de establecer la conexión a un punto de acceso (Router). Al principio, la conexión la hacía correctamente pero llego un momento en el que no llegaba a unir. Investigando se llego a la conclusión de que, había que reinicializar el módulo, resetearlo para borrar posibles valores defectuosos que se quedaran grabados en la memoria.

Si hablamos de las horas trabajas e invertidas en cada grupo, este diagrama muestra la proporción dedicada a cada uno de ellos:

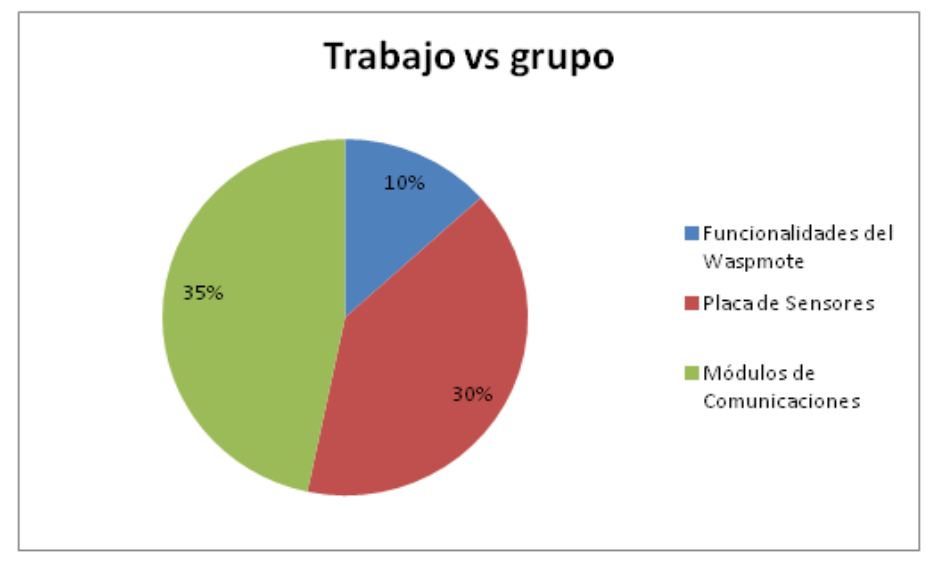


Fig. 6.1. Diagrama de sectores del trabajo dedicado en cada grupo

Si hablamos del tiempo dedicado en cada etapa para el desarrollo del trabajo, este diagrama muestra la proporción dedicada en cada uno de ellos:

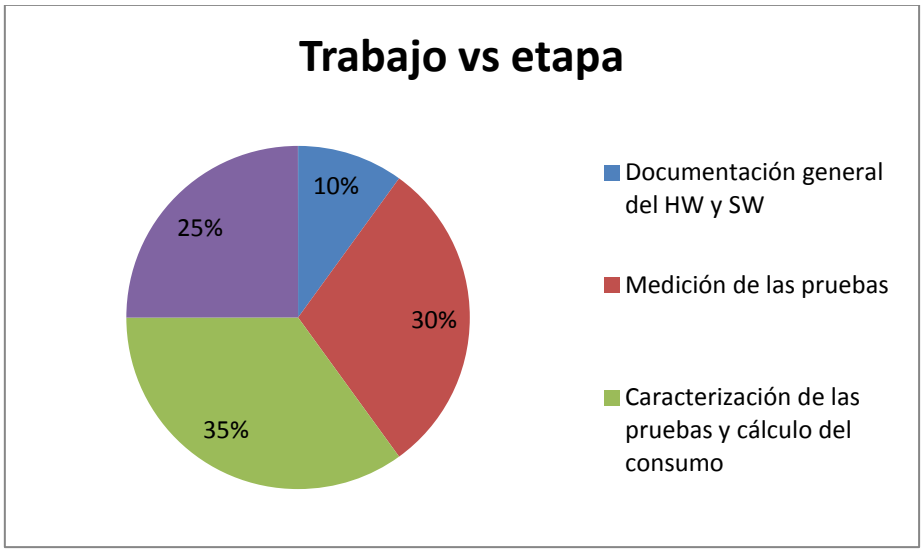


Fig. 6.2. Diagrama de sectores del trabajo dedicado en cada etapa

7. Conclusiones.

7.1. Cumplimiento e incumplimiento de objetivos

Se cubrió satisfactoriamente tanto el objetivo principal del proyecto, que era caracterizar el consumo eléctrico de los distintos modos y opciones de funcionamiento de la plataforma Waspmote incluyendo sus sensores y módulos de comunicaciones asociados, a través de medidas experimentales para poder realizar estimaciones de autonomía de su batería de alimentación realizando lo que se conoce como "Energy Guide", cada vez de mayor relevancia en productos electrónicos de todo tipo y, especialmente, en redes sensoriales inalámbricas, como toda la serie de objetivos parciales comentados en el apartado 1 de la memoria, sin dejar de incumplirse ninguno de ellos.

Este proyecto, llega a ser innovador ya que no se conoce algo parecido en plataformas de sensores y ambicioso, ya que se pretende caracterizar todo el ecosistema. El ámbito es todo Waspmote.

Se ha calculado con precisión el tiempo de vida del Waspmote en función de las acciones que queramos ejecutar, y del ciclo de trabajo que hemos querido imponer.

Cada consumo es sumado al que tienen los demás dispositivos que trabajan con él: filosofía modular.

Se ha intentado que todo sea lo más cercano a la realidad, al uso que le dará el cliente.

Para crear el código que ejecuta Waspmote para medir el consumo, siempre se parte de uno de los ejemplos de código presentes en la web: se ha usado el más limpio y eficiente que se ha podido, eliminando acciones no necesarias que añadirían consumo irreal.

Se ha descubierto una serie de *bugs* sea a nivel de HW, como SW.

Se han incidido en las acciones más costosas, ya sea por tiempo de ejecución o por alto consumo eléctrico, o ambos.

La calculadora de consumos realizada llega a ser muy novedosa, automática y de gran

utilidad para el cliente, ya que es una aplicación software que presenta varias opciones para el usuario y en base a las seleccionadas, puede calcular los meses o años de vida que tendrá el dispositivo.

Esta calculadora no sólo tiene en cuenta las pérdidas energéticas, sino los aportes energéticos (inyección de corriente desde el panel solar o desde el puerto USB)

Como conclusión personal, diré que, durante este año que he pasado en Libelium, he aprendido mucho de toda esta tecnología, de su magnífica plantilla, ya que, gracias a ellos, hemos podido formarnos muy bien de todo lo que forman las redes sensoriales inalámbricas, de cómo se utilizan, de cómo trabajan cada día para seguir innovando y mejorando y que, también, me han servido de gran ayuda en todas las dudas que me iban surgiendo y, además, me lo he pasado muy bien.

7.2. Consejos" para reducir el consumo y líneas futuras

Uno de los problemas más importantes es el consumo de energía de los nodos. Para lograr que éste sea mínimo y, por tanto, conseguir un máximo tiempo de vida de la red habrá que tener en cuenta estos puntos:

- La comunicación de mensajes es el primer consumidor de energía.
- La CPU puede quedarse en un estado sleep de bajo consumo mientras no tenga que procesar ni enviar nada.
- Economizar la distancia de las comunicaciones.
- Técnicas de software: programación eficiente de líneas de código.

A pesar de que las baterías que utilizamos, siendo la principal fuente de energía de estos motes, son recargables, debemos tener en cuenta todos estos puntos, para que la batería aguante lo máximo posible y, así, no estar recargándolas cada dos por tres.

Como hemos comentado en el anterior apartado, se ha eliminado las acciones no necesarias en el código utilizado para hacerlo lo más eficiente posible, y sabiendo el usuario, el consumo de cada opción medida, podrá saber qué códigos son más eficientes que otros, con lo cual, un punto a mejorar es, diseñar a nivel de software, un código que mejore al anterior, optimizarlo y, así, ser mucho más eficiente, o si el hardware diseñado no da un consumo óptimo de energía, mejorarlo para tal fin.

El consumo de energía viene dado por lo que consumen los nodos y opciones de funcionamiento de la plataforma Waspmote, de los sensores, de la comunicación y del procesado. La mayor cantidad de energía es consumida en la transmisión de información, siendo menor en el procesado y uso de los sensores, por eso mismo hay que evitar que el usuario este midiendo cada acción muy a menudo, para que el consumo sea menor y evitar que la batería les aguante escasos días. Por ejemplo, no tiene sentido medir la temperatura muy a menudo (como cada 10 segundos) y enviar por XBee, aunque muchos clientes lo hacen y luego se sorprenden de que la batería les aguante 5 días. Por lo tanto, hay que evitar los malos usos.

Si a la batería le queda poca carga, lo que se debería es evitar ejecutar acciones o procesos que requieran de un gran consumo. Si utilizando o usamos procesos que requieran un consumo mínimo de energía, la batería aguantará más tiempo.

También es importante que evitemos medir consumos de acciones con el dispositivo encendido todo el rato, ya que eso requiere una gran pérdida de consumo y la batería aguantaría menos de lo debido. Se aconseja hacer procesos completos, es decir, encender el dispositivo y hacer las demás acciones que se requieran y una vez hecho eso, llevarlo a dormir.

Respecto a los módulos de comunicación, en lo que se refiere al envío de paquetes, se puede reducir el consumo, reduciendo el número de bytes de carga útil o reduciendo el paquete de datos. Llevarlo al máximo requiere más consumo que si se reduce.

Los módulos de comunicación, al ser el grupo que más consume, se debe de evitar realizar acciones repetitivas ya que chupan mucha energía y puede producir que la batería se agote en escasos días.

Otro punto a tener en cuenta sobre los módulos es que en el envío de los paquetes en modo cifrado requiere más consumo. Es un modo más seguro de envió pero, si se requiere reducir su consumo, se recomendaría reducir este tipo de envío y hacerlo sin cifrar ya que requiere menos consumo que el primero.

En algunas ocasiones, cuando estamos midiendo el consumo energético en la transmisión y recepción de un paquete en un módulo de comunicación, al estar tanto el emisor y el receptor a poca distancia entre ellos, ocurre que puede llegar a haber hasta cuatro reintentos para poder transmitirlo, y eso conlleva mas consumo de energía del necesario, con lo cual para evitar esto, lo que se podría hacer es tenerlos a una distancia considerable para que no pase tal caso, y evitar consumos innecesarios.

Alguna de las técnicas empleadas para minimizar el consumo de energía en el procesado de datos incluye reducir el voltaje de alimentación y disminuir la frecuencia de trabajo del microprocesador durante períodos de baja actividad.

Reducir el voltaje de alimentación es un método muy eficaz para reducir el consumo de energía en el estado activo. Por otro lado, cuando un microprocesador maneja una carga computacional que varía con el tiempo, reducir la frecuencia de trabajo durante los períodos de actividad reducida causa igualmente una disminución del consumo de energía. A su vez, esto compromete el rendimiento del procesador. Ahorros de energía significativos puede provocar que el rendimiento máximo no sea siempre el deseado y por lo tanto, el voltaje de alimentación y la frecuencia de trabajo deben ser adaptadas a las exigencias de procesamiento en cada momento.

Se ha hecho, todas las acciones medidas, ya que viene así por defecto, a la máxima potencia (ya que hay que ponerse en el peor caso). Eso conlleva mayor consumo de energía, con lo cual se podría mejorarlo, reduciendo la potencia.

Actualmente se están estudiando sistemas basados en energías renovables para solucionar el problema de la energía en estos nodos, basados en energía solar, termogeneración, energía basada en vibraciones, etc...

Aunque la caracterización de los módulos es buena, se podría hacer un estudio más a fondo y analizar más en profundidad las curvas obtenidas en el osciloscopio sin realizar aproximaciones.

Diseñar un nuevo código más eficiente para reducir el consumo en algunos dispositivos que requieran mayor consumo energético.

Mejorar el HW en algunas placas de sensores, ya que cuando se apagaban las placas, el consumo no bajaba a 15.6 mA que es el consumo normal cuando el Waspmote está en On, si no que se quedaba en 16 mA. Esa diferencia de 0.4 mA puede que a ojos nuestros sea mínima, pero a ojos del dispositivo y de la batería no ya que eso se nota y puede producir mayores pérdidas de consumo de lo normal.

8. Referencias.

[1] Wireless Sensor Networks: Technology & Applications. K. Pister. University of Berkeley.

[2] http://en.wikipedia.org/wiki/Wireless_sensor_network

[3] Las tecnologías de la información en la agricultura, una asignatura pendiente. B. Recio, C. Valero y B.Diezma. Dossier en la revista Vida Rural, numero 293, julio de 2009.

[4] Commercial- and Industrial-Class Wireless Sensor Networks. Millenial Net. 2005.

[5] http://riunet.upv.es/bitstream/handle/10251/39371/Lajara

[6] http://www.mfbarcell.es/conferencias/wsn.pdf

[7] Wireless Sensor Networks: Estado del Arte e Investigación. M. Soledad Escolar Díaz.

[8] http://blogs.heraldo.es/ciencia/files/2010/08/waspmote_400.png

[9] http://riunet.upv.es/bitstream/handle/10251/8592/PFC%20-%20DESARROLLO%20DE%20APLICACIONES%20BASADAS%20EN%20WSN.pdf.txt

[10] http://es.rs-online.com/web/p/adaptadores-para-pinzas-amperimetricas-demultimetros/3944997/

[11] http://www.libelium.com/downloads/documentation/waspmote_technical_guide.pdf

[12] http://www.libelium.com/downloads/documentation/waspmote_programming_guide.pdf

[13] http://www.libelium.com/downloads/documentation/waspmote_ide_user_guide.pdf

[14] http://diy.bq.com/wp-content/uploads/2014/06/CuteCom.png

[15] http://bibing.us.es/proyectos/abreproy/12046/fichero/4_Capitulo4.pdf

[16] http://www.libelium.com/uploads/2013/08/quickstart_guide.pdf

[17] http://www.libelium.com/downloads/documentation/agriculture_sensor_board_2.0.pdf

[18] www.libelium.com/downloads/documentation/events-sensor-board_2.0.pdf

[19] http://www.libelium.com/downloads/documentation/gases_sensor_board_2.0.pdf

[20] http://www.libelium.com/downloads/documentation/smart_parking_sensor_board.pdf

[21] http://www.libelium.com/downloads/documentation/smart_metering_sensor_board_2.0.pdf

[22] http://www.libelium.com/uploads/2013/02/prototyping-sensor-board_2.0_eng.pdf

[23] http://www.libelium.com/downloads/documentation/smart_cities_sensor_board.pdf

[24] http://www.libelium.com/downloads/documentation/radiation_board.pdf

[25] http://www.libelium.com/downloads/documentation/smart_water_sensor_board.pdf

[26] http://www.libelium.com/downloads/documentation/video_camera_guide.pdf

[27] http://www.libelium.com/downloads/documentation/waspmote-802.15.4-networking_guide.pdf

[28] http://www.libelium.com/downloads/documentation/waspmote-868-networking_guide.pdf

[29]http://www.libelium.com/downloads/documentation/waspmote-digimesh-networking_guide.pdf

[30] http://www.libelium.com/downloads/documentation/waspmote-zigbee-networking_guide.pdf

[31] http://www.libelium.com/downloads/documentation/wifi_networking_guide.pdf

[32] http://www.libelium.com/downloads/documentation/bluetooth-device-networking_guide.pdf

[33] http://www.libelium.com/downloads/documentation/bluetooth-low-energy-networking_guide.pdf

[34]http://www.libelium.com/downloads/documentation/waspmote_gsm_gprs_pro_networking_guide.pdf

[35] http://www.libelium.com/downloads/documentation/3G_GPS_guide.pdf

[36] http://www.libelium.com/downloads/documentation/waspmote-gps-programming_guide.pdf

Anexo A: Dispositivo Waspmote.

Waspmote [15] se basa en una arquitectura modular. La idea es integrar únicamente los módulos que necesitemos en cada dispositivo y ser capaces de cambiarlos y ampliarlos según las necesidades.

Los módulos disponibles para integrar en Waspmote se clasifican en:

- Módulos ZigBee/802.15.4 (2.4GHz, 868MHz, 900MHz). Baja y alta potencia.
- Módulo GSM.
- 3G/GPRS (Quadband: 850MHz/900MHz/1800MHz/1900MHz).
- Módulo GPS.
- Módulos Sensoriales (Placas de Sensores).
- Módulo de almacenamiento: SD Memory Card.

En la siguiente lista podemos ver las especificaciones del dispositivo:

- Microcontrolador: ATmega1281
- Frecuencia: 14.7456MHz
- SRAM: 8KB
- EEPROM: 4KB
- FLASH: 128KB
- SD Card: 2GB
- Peso: 20gr
- Dimensiones: 73.5 x 51 x 13 mm
- Rango de Temperatura: [-10ºC, +65ºC]

Las siguientes figuras muestran los componentes principales en *Waspmote:*

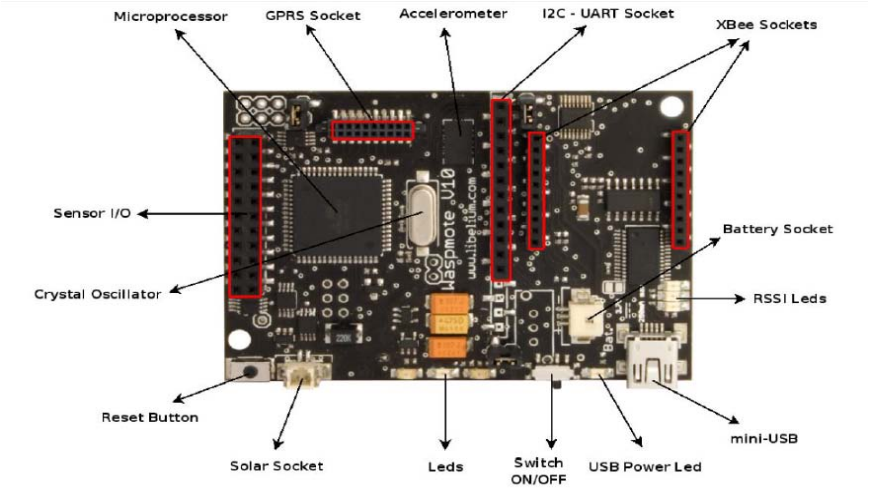


Fig. A.1. Cara superior del Waspmote y los sockets disponible [15]

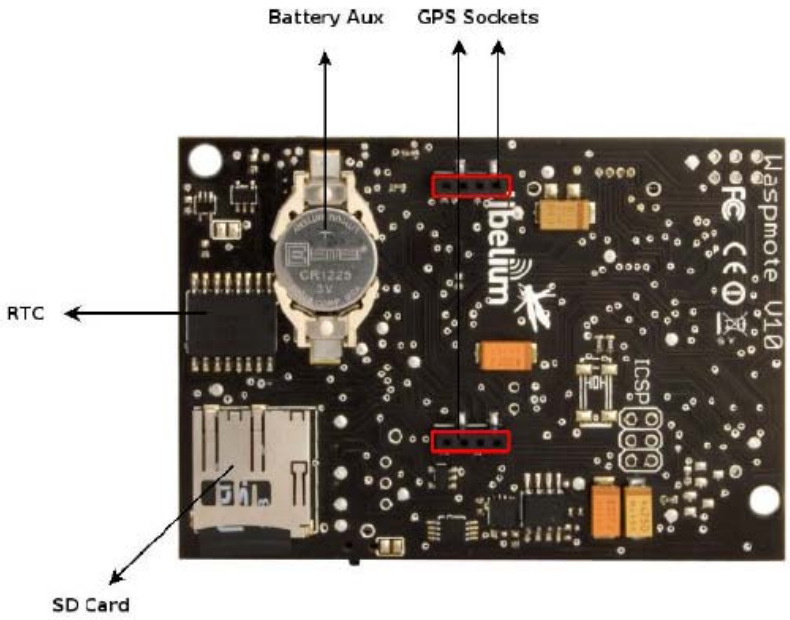


Fig. A.2. Parte inferior del Waspmote y sockets disponibles [15]

En las fig. A.3 y A.4 podemos ver diagramas de bloques del dispositivo Waspmote. El diseño del dispositivo Waspmote fue realizado por el equipo de Libelium con anterioridad a la realización de este PFC.

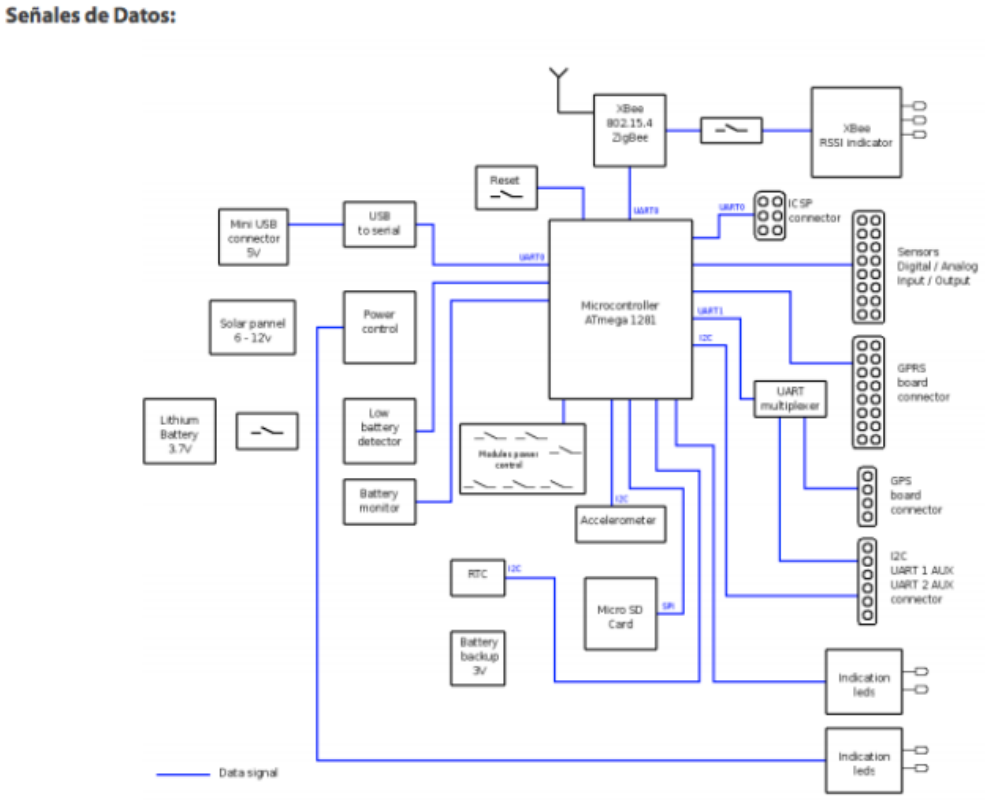


Fig. A.3. Señales de datos [15]

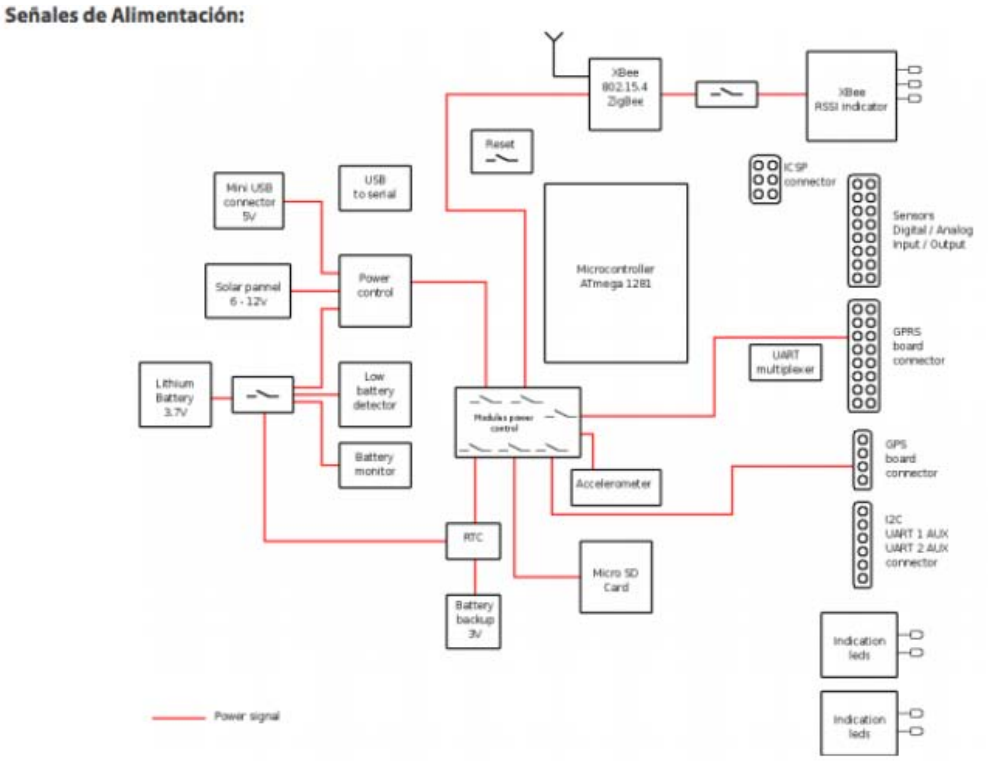


Fig. A.4. Señales de alimentación [15]

El dispositivo está basado en un microcontrolador ATMEGA 1281 que funciona a 8MHz. La ventaja de los microcontroladores ATMEGA es que se pueden utilizar en un entorno amigable para el usuario. Para ello, es necesario cargar un '*bootloader*' antes de empezar a programarlo, que se encargará de cargar el programa a nuestro microcontrolador.

El *bootloader* es el programa que se encarga de preparar al microcontrolador para que acepte el código de cada programa que queremos cargar. De esta forma, la utilización de un dispositivo externo como un programador sólo es necesaria para la carga de dicho *bootloader*. Una vez cargado el *bootloader* se podrá hacer uso de un *IDE (Integrated Development Environment)* para cargar los programas al dispositivo.

En cuanto a consumo, Waspmote tiene 4 modos de funcionamiento:

- ON: modo normal de funcionamiento. El consumo en este estado es de 9mA.
- Sleep: El programa principal se detiene, el microcontrolador pasa a un estado de latencia, del que puede ser despertado por todas las interrupciones asíncronas y por la interrupción síncrona generada por el Watchdog. El intervalo de duración de este estado va de 32ms a 8s. El consumo en este estado es de 62µA.
- Deep Sleep: El programa principal se detiene, el microcontrolador pasa a un estado de latencia del que puede ser despertado por todas las interrupciones asíncronas y por la interrupción síncrona lanzada por el RTC. El intervalo de este ciclo puede ir de 8 segundos a minutos, horas, días. El consumo en este estado es de 62µA.
- Hibernate: El programa principal se detiene, el microcontrolador y todos los módulos de Waspmote quedan completamente desconectados. La única forma de volver a activar el dispositivo es a través de la alarma previamente programada en el RTC (interrupción síncrona).

La placa cuenta con un conector mini-USB para comunicar el microcontrolador con un ordenador, de forma que se pueda programar a través de un entorno amigable. Este conector también nos permite la captura presentación por pantalla de los datos generados por el dispositivo *Waspmote*.

La alimentación puede ser introducida a través del mini-USB, mediante una placa solar o bien a través de una batería de litio. La batería de litio se puede recargar al tener el módulo conectado al mini-USB o a la placa solar. De esta forma, podemos recargar el dispositivo *Waspmote* mientras lo programamos, o incrementar la duración de la vida de su batería al tenerlo conectado a una placa solar.

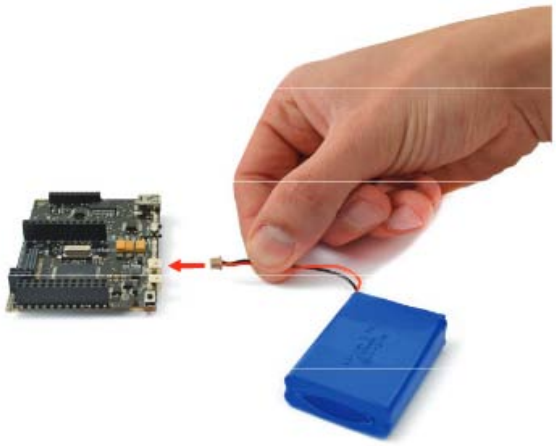


Fig. A.5. Dispositivo Waspmote con batería de litio [16]

El dispositivo *Waspmote* dispone de 10 E/S digitales para poder conectar todo tipo de sensores externos, así como de 8 entradas analógicas. De esta forma obtenemos una gran cantidad de posibilidades de interconexión para nuestras redes sensoriales. En la siguiente figura se muestra como ejemplo el sensor de radiación:

Fig. A.6. Sensor de radiación para Waspmote [11]

El dispositivo *Waspmote* también cuenta con un botón de reset, que permite reiniciar el código cargado en el microcontrolador. El código se reiniciará desde el punto de partida, siendo este código el último que se haya cargado en la placa.

La transmisión de datos se puede realizar de varias maneras. Una de ellas es a través de los módulos de comunicación implementados (el módulo Wi-Fi, o módulos ZigBee, Bluetooth,… ya implementados). La otra forma de transmitir es mediante 3G.

Fig. A.7. Módulos ZigBee, Bluetooth y Wi-Fi para Waspmote. [11]

El almacenamiento de los datos capturados se puede realizar en una tarjeta mini- SD. Para ello, se ha diseñado un zócalo de conexión de dicho tipo de tarjetas SD. De esta forma, se pueden almacenar gran cantidad de datos, concretamente se puede trabajar con tarjetas de hasta 4 Gbytes.

La sincronía y funciones de reloj se pueden realizar gracias al uso de un reloj interno. Dicho reloj sirve para poder tener la red sincronizada en caso de ser necesario, pudiendo despertar al dispositivo de un estado de bajo consumo.

Para posibles aplicaciones móviles o detectar caídas del dispositivo, se ha dotado al dispositivo Waspmote con un acelerómetro. De esta forma, se puede detectar si el dispositivo ha sufrido una caída o ha cambiado de dirección en un movimiento constante.

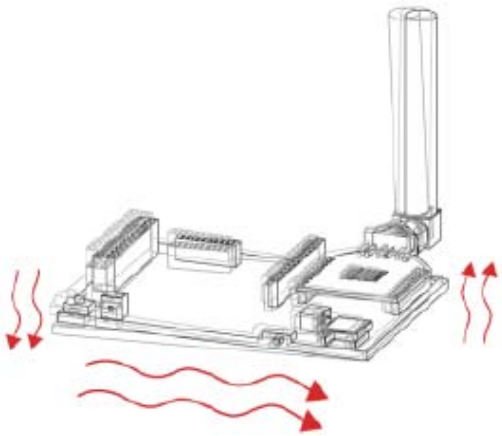


Fig. A.8. Uso del acelerómetro frente a vibraciones [11]

Finalmente, con el fin de dotar a dichos dispositivos de un sistema de localización, se ha diseñado una placa de interconexión para conectar un receptor GPS. De esta forma, podemos obtener la localización en cada momento de los dispositivos *Waspmote*.

Anexo B: Hardware adicional utilizado para el cálculo del consumo energético de Waspmote.

En este apartado, se va a explicar más ampliamente el funcionamiento de todas las funcionalidades, placas de sensores y módulos de comunicaciones que integran la plataforma Wasmote y que se han utilizado para el cálculo del consumo en cada uno de ellos, explicados resumidamente en el apartado 2 de la memoria.

B.1. Funcionalidades de la plataforma Waspmote.

B.1.1. Leds [11]

- Led 0: Es un indicador de color rojo. Está conectado al microcontrolador. Es totalmente programable por el usuario desde el código de programa. Además, el LED 0 indica cuando Waspmote se resetea, parpadeando cada vez que un reinicio se lleva a cabo en la placa.
- Led 1: Es un indicador de color verde. Está conectado al microcontrolador. Es totalmente programable por el usuario desde el código de programa.

B.1.2. Memoria EEPROM [11]

Para almacenar de forma permanente los valores, es necesario utilizar la EEPROM del microcontrolador (4 KB) de memoria no volátil. Las Direcciones EEPROM de 0 a 1023 son utilizados por Waspmote para guardar los datos importantes, por lo que no deben ser sebrescritos. Por lo tanto, las direcciones de almacenamiento disponibles van de 1024 a 4095.

B.1.3. Real Time Clock o RTC [11]

Waspmote tiene un reloj de tiempo real o RTC, que lo mantiene informado del tiempo. Esto permite a Waspmote a ser programado para realizar acciones relacionadas con el tiempo, tales como sleep, hibernate, deep sleep, wake up, alarmas, entre otros. Está alimentado por la batería.

- Temperatura: El RTC de Waspmote tiene un sensor de temperatura interno construido que se utiliza para recalibrarse a sí mismo. Waspmote puede acceder al valor de este sensor a través del bus I2C.

B.1.4. Sistemas de Energía [11]

- Sleep: El programa principal se detiene, el microcontrolador pasa a un estado de latencia, del que puede ser despertado por todas las interrupciones asíncronas y por la interrupción síncrona generada por el Watchdog. El intervalo de duración de este estado va de 32ms a 8s. El consumo en este estado es de 62µA.

- Deep Sleep: El programa principal se detiene, el microcontrolador pasa a un estado de latencia del que puede ser despertado por todas las interrupciones asíncronas y por la interrupción síncrona lanzada por el RTC. El intervalo de este ciclo puede ir de 8 segundos a minutos, horas, días. El consumo en este estado es de 62µA.

- Hibernate: El programa principal se detiene, el microcontrolador y todos los módulos de Waspmote quedan completamente desconectados. La única forma de volver a activar el dispositivo es a través de la alarma previamente programada en el RTC (interrupción síncrona).

- Battery: Sirve como fuente de alimentación al sistema. Es recargable y tu tensión oscila entre 3.7V y 4.2V.

B.1.5. Tarjeta de memoria micro-SD [11]

Waspmote tiene soporte de almacenamiento externo, como las tarjetas SD (Secure Digital). Estas tarjetas micro-SD se utilizan específicamente para reducir espacio en la placa a un mínimo.

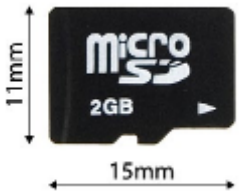


Fig. B.1. Tarjeta micro-SD

B.1.6. Acelerómetro [11]

Waspmote ha incorporado un sensor de aceleración que informa de las variaciones de aceleración del mote experimentada en cada uno de los 3 ejes (X, Y, Z). La integración de este sensor permite la medición de la aceleración en los 3 ejes (X, Y, Z), estableciendo 4 tipo de eventos: Caída Libre, despierte inercial, el movimiento 6D y la posición 6D.

B.2. Placas Integradas con sus respectivos sensores.

B.2.1. Placa de Agricultura

La placa de Agricultura [17] de Waspmote 2.0 permite controlar múltiples parámetros ambientales involucrando una amplia gama de aplicaciones, desde el análisis de desarrollo de cultivo hasta la observación meteorológica. Para ello, se ha dotado de 15 sensores, los cuales se pueden conectar al mismo tiempo. Con el objetivo de extender la durabilidad del dispositivo después de la utilización, la placa está dotada de un sistema de detectores de estado sólido que facilita una regulación precisa de su energía, lo que prolonga la vida de la batería.

La placa de agricultura incluye toda la electrónica y sockets necesarios para conectar los sensores típicos:

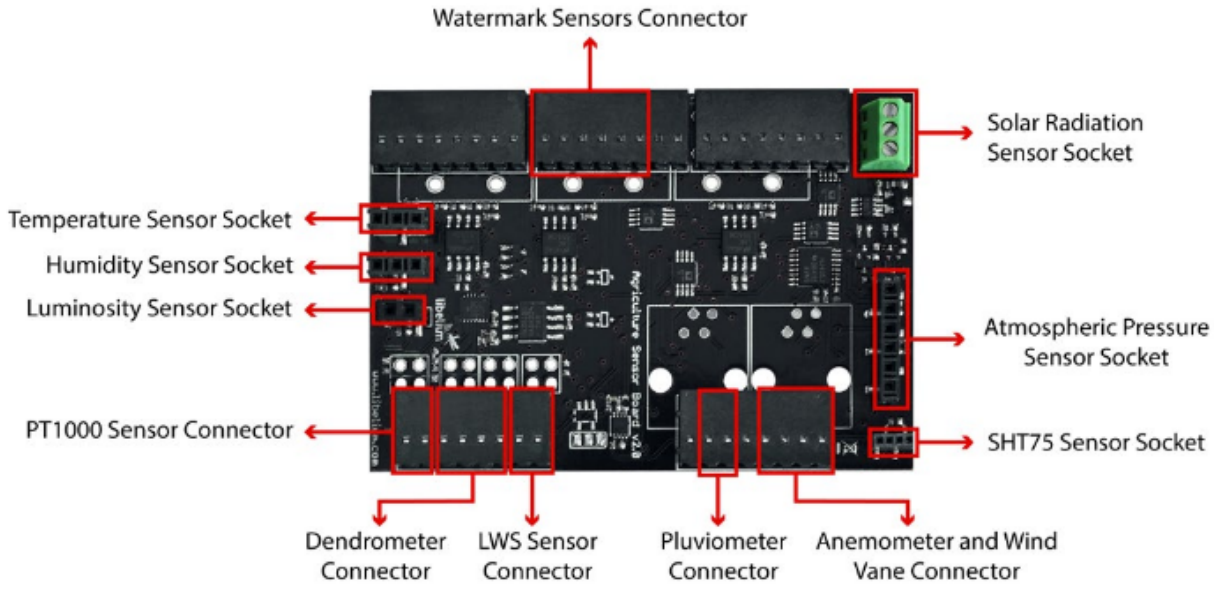


Fig. B.2. Cara superior de la placa de Agricultura con sus correspondientes sensores [17]

B.2.2. Placa de Eventos

La placa de Eventos [18] de Waspmote 2.0 permite controlar múltiples parámetros involucrando una amplia gama de aplicaciones como la seguridad para puertas y ventanas, detección de personas, emergencias tales como detección de nivel de agua, temperatura, presencia o el control de productos en logística. Para ello se han dotado de 12 sensores, como presión, vibración, nivel de líquido, luminosidad, fuerza, humedad entre otros.

La placa de eventos incluye toda la electrónica y sockets necesarios para conectar los sensores típicos:

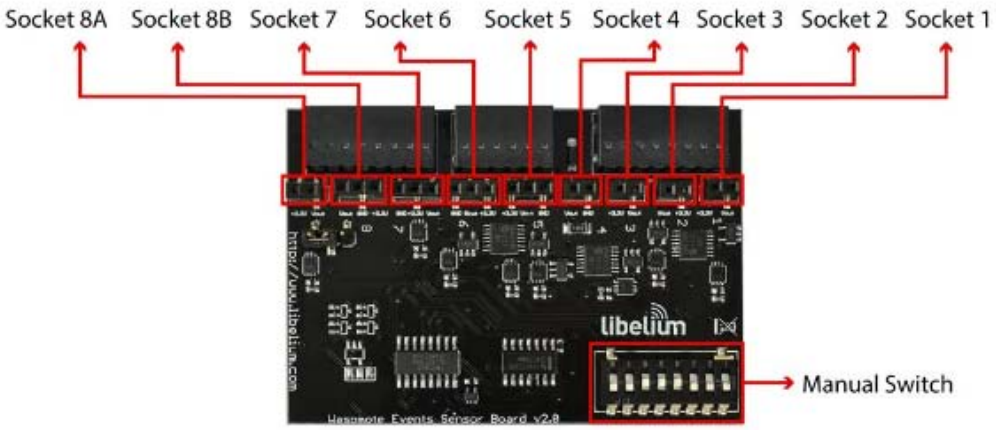


Fig. B.3. Cara superior de la placa de Eventos [18]

Como se ve en la imagen, la placa dispone también de 8 interruptores manuales correspondientes a cada uno de los sockets que se activarán y controlaran la energía de cada sensor.

B.2.3. Placa de Gases

La placa de gases [19] ha sido diseñada para controlar los parámetros ambientales tales como temperatura, humedad, presión atmosférica y 14 tipos diferentes de gases como CO, CO_2, O_2, NH_3, entre otros. Permite la inclusión de 7 sensores de gases, al mismo tiempo, la regulación de su energía a través de un sistema de conmutadores de estado sólido y la amplificación de la señal de salida de cada uno de ellos a través de una etapa de amplificación no inversora de una ganancia máxima de 101 controlada por un potenciómetro digital configurable a través del bus de circuito integrado I2C.

La placa de gases incluye toda la electrónica y sockets necesarios para conectar los sensores que correspondan en cada uno de ellos:

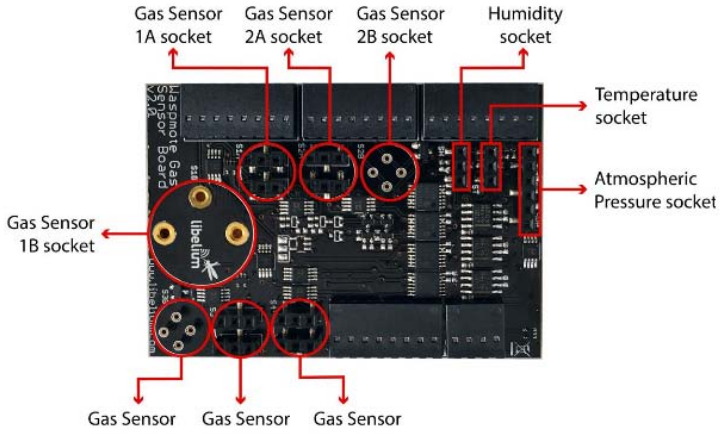


Fig. B.4. Cara superior de la placa de Gases [19]

B.2.4. Placa de Parking

La placa de parking [20] se basa en la variación del campo magnético por encima de la placa causada por el chasis de un vehículo colocado en el estacionamiento donde el mote se ha utilizado. Este cambio se detecta utilizando un sensor de campo magnético (MFS), un material cuya resistencia eléctrica varía con el campo magnético a través de él, medido y procesado para proporcionar el sistema con una respuesta que se puede utilizar para determinar el estado de la plaza de aparcamiento. Por lo tanto tiene una serie de aplicaciones, como la detección de coches para la información sobre el estacionamiento disponible, detección de placas de aparcamiento libres y para el control de terrenos de estacionamientos paralelos y perpendiculares.

La placa de parking incluye toda la electrónica y socket necesario para conectar el sensor correspondiente:

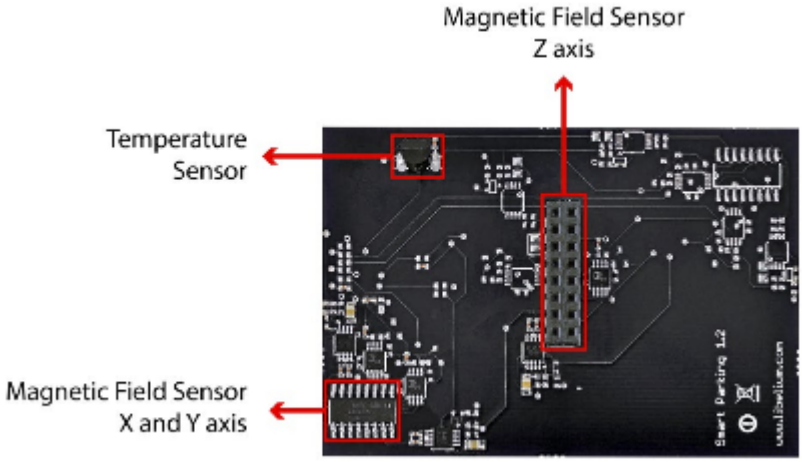


Fig. B.5. Cara superior de la placa de Parking [20]

Fig. B.6. Sensor del eje Z [20]

B.2.5. Placa de Smart Metering

La placa de Smart Metering [21] se ha concebido para controlar los parámetros que pueden requerir para ser controlados en un entorno doméstico. Incluye sensores para la alimentación y el control del consumo de agua, el desplazamiento, la luminosidad y la

humedad ambiental. En la versión normal de la placa, hasta 8 sensores, alimentados de forma independiente, se pueden conectar al mismo tiempo, mientras que existe una versión PRO con la electrónica necesaria para el control de la célula de carga y la adaptación.

La placa de gases incluye toda la electrónica y sockets necesarios para conectar los sensores que correspondan en cada uno de ellos:

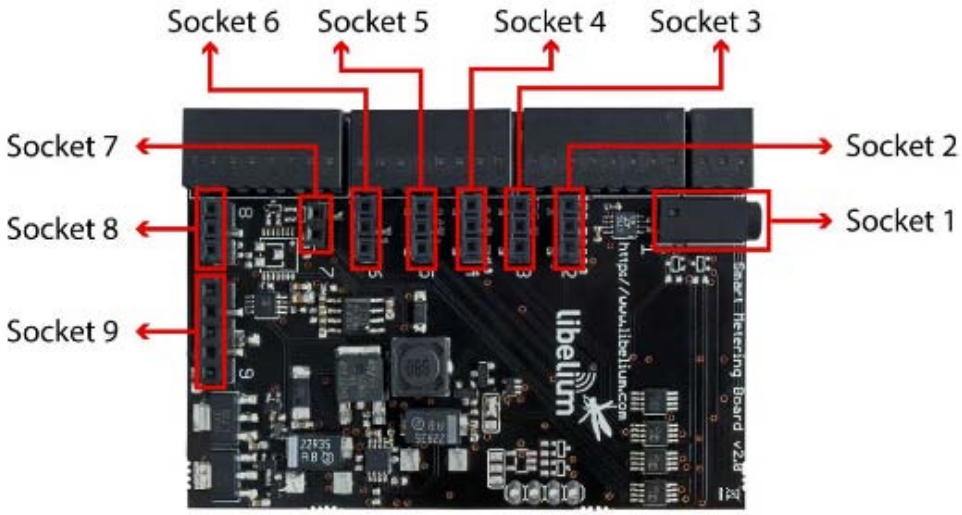


Fig. B.7. Cara superior de la placa de Smart Metering [21]

B.2.6. Placa de Prototipado

La placa de prototipado [22] ha sido diseñado para que sea lo más fácil posible para el usuario para integrar cualquier tipo de sensor. Con este objetivo en mente, la placa ha sido dotada de un convertidor analógico-digital de 16 bits (ADC), que proporciona hasta una resolución 68uV en un rango de 0 a 4,5 V para una entrada diferencial; un área de pads independientes donde se pueden soldar pins, cables, circuitos integrados encapsulados pasivos o DIP; y un área para circuitos integrados encapsulados SMD sobre los cuales, se pueden montar varios circuitos o sockets con diferentes tamaños.

La placa de prototipado incluye toda la electrónica y sockets necesarios para tales fines:

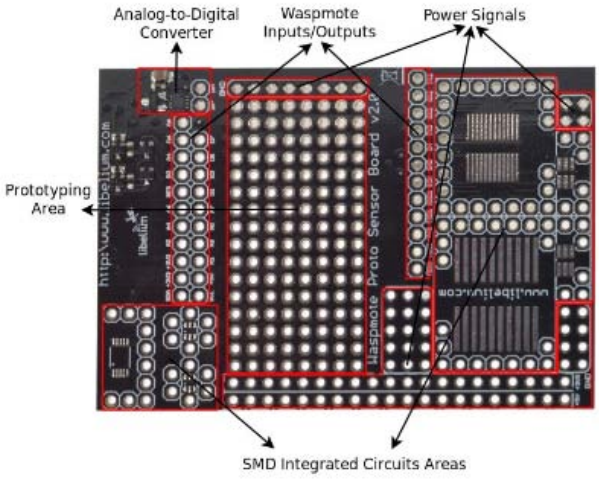


Fig. B.8. Cara superior de la placa de prototipado [22]

B.2.7. Placa de Smart Cities

El propósito de la placa de Smart Cities [23] es ampliar las funcionalidades de control de la placa de Smart Metering desde ambientes interiores a lugares al aire libre. Además de los sensores de humedad, luminosidad y temperatura, presentes en todas las placas de Libelium, se han incluido otros dos sensores para aplicaciones específicas: tres sensores destinados a controlar las grietas en los edificios y estructuras, un sensor de desplazamiento lineal de ancho de rotura y un medidor de deformación para la detección de fisuras. También, se han introducido un sensor de polvo y partículas, que se utiliza para medir la concentración de partículas en suspensión en el medio ambiente en aplicaciones de control de calidad del aire, y finalmente un micrófono, adaptado para medir el ruido ambiental en la escala en decibelios.

La placa de gases incluye toda la electrónica y sockets necesarios para conectar los sensores que correspondan en cada uno de ellos:

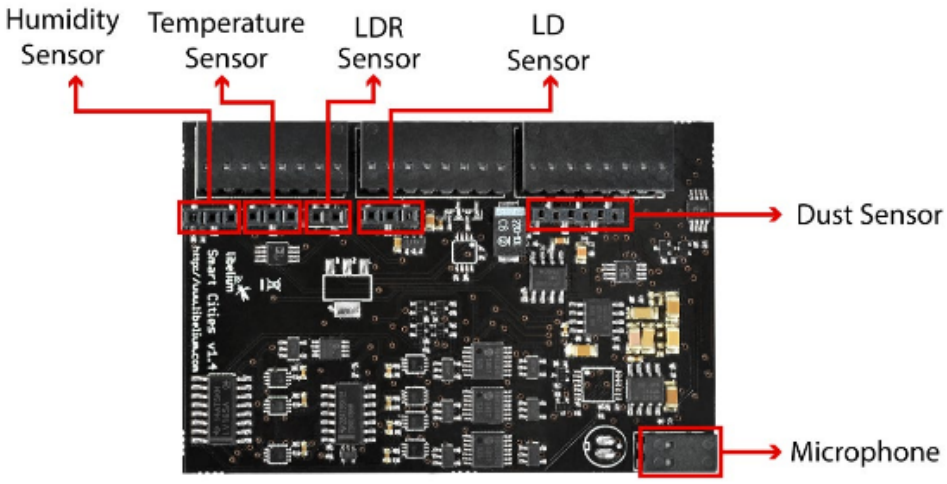


Fig. B.9. Cara superior de la placa de Smart Cities con sus correspondientes sensores [23]

B.2.8. Placa de Radiación

La placa del sensor de radiación [24] se basa en un tubo Geiger-Müller. La mayoría de los detectores incluyen un amplificador de audio que produce un chasquido audible en la descarga. El número de pulsos por segundo mide la intensidad del campo de radiación. Algunos contadores Geiger muestran una tasa de exposición (por ejemplo mR·h), pero esto no se relaciona fácilmente con una tasa de dosis como el instrumento no discrimina entre la radiación de diferentes energías.

Esta placa tiene una serie de aplicaciones como controlar los niveles de radiación de forma inalámbrica sin comprometer la vida de las fuerzas de seguridad; crear redes de radiación de prevención y control de una planta nuclear o medir la cantidad de radiaciones Beta y Gamma en áreas específicas de forma autónoma.

Las partes de la placa de radiación se muestran en la siguiente imagen:

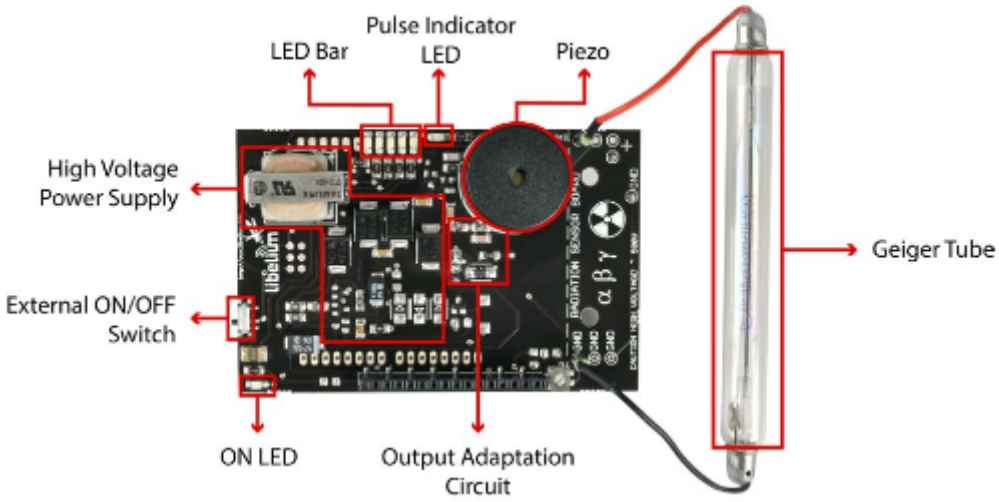


Fig. B.10. Cara superior de la placa de radiación [24]

B.2.9. Placa de Smart Water

La placa de Smart Water [25] ha sido diseñado para facilitar la medición de los parámetros químicos más importantes que permiten el control remoto de la calidad del agua en diferentes escenarios, que incluye la vigilancia de la contaminación en ambientes naturales, tales como ríos y lagos, el control de las condiciones adecuadas de agua en piscinas o granjas de peces y la observación de las aguas residuales industriales de industrias. Entre estos parámetros se incluyen la temperatura del agua, conductividad, pH, oxígeno disuelto, potencial de oxidación-reducción (ORP), diferentes iones disueltos y turbidez.

La placa de prototipado incluye toda la electrónica y sockets necesarios para tales fines:

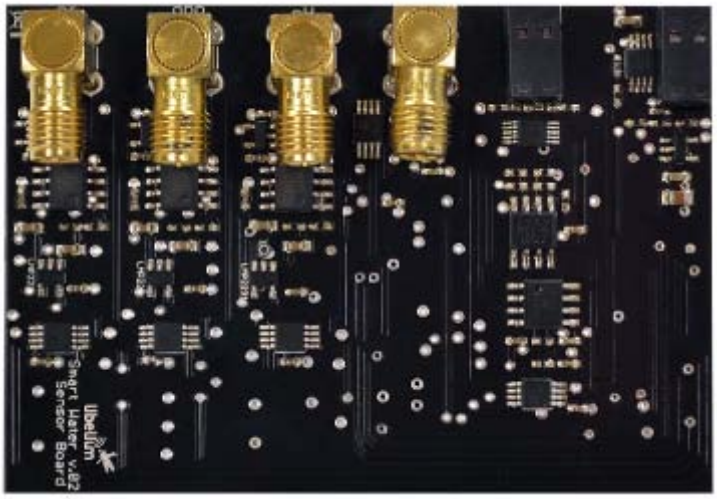


Fig. B.11. Cara superior de la placa de Smart Water [25]

B.2.10. Placa Video Cámara

La nueva placa de video cámara [26] en combinación con el módulo 3G permite tomar fotos (VGA - 640x480) y grabar vídeos (QVGA - 320x240) y enviarlos a la nube mediante el uso de WCDMA de alta velocidad y redes de telefonía móvil HSPA de la misma forma que lo hacen los teléfonos inteligentes. Esto hace posibles que nodos sensores envíen no solo información de sensores discretos, como la temperatura o la humedad (que puede ser codificado usando un solo número), sino también strems complejos de información, como fotos y vídeos. Esta nueva característica permite a los desarrolladores la creación de nuevas aplicaciones de seguridad inteligente.

La placa de Video Cámara permite a Waspmote a tomar fotos y grabar video, junto con el módulo 3G/GPRS. La placa incluye 22 LEDs IR, dividido en dos bloques. Cada uno está controlado por transistores, para dar iluminación adicional y grabar con poca luz o en la noche. Para eliminar la distorsión de IR cuando la placa se utiliza con la luz natural, la placa tiene un intercambiador de filtro con un filtro de luz IR.

La placa tiene dos sockets para un un fotodiodo IR y LDR. Con la información de estos sensores, los usuarios pueden seleccionar el filtro adecuado y, si es necesario, utilizar los LEDs IR.

La placa de Video Cámara, también incluye un sensor de presencia PIR para generar una interrupción en Waspmote y tomar una foto o grabar un vídeo cuando una persona pasa. Esta función está especialmente diseñada para aplicaciones de seguridad y vigilancia.

Fig. B.12. Cara superior de la placa de Video Cámara [26]

B.3. Módulos de Comunicaciones.

B.3.1. XBee 802.15.4

IEEE-802.15.4-2006 [27] es un estándar establecido por el IEEE (*Institute of Electrical and Electronics Engineers*) para redes de comunicaciones LR-**WPAN** (*low-rate wireless personal area networks*). Es decir, sirve para realizar las transmisiones entre maquinas en aplicaciones de baja velocidad, bajo coste y bajo consumo. Por sus características, es un estándar ampliamente usado en las redes sensoriales distribuidas.

Fig. B.13. XBee 802.15.4 PRO [27]

Los módulos XBee 802.15.4 cumplen con el estándar IEEE 802.15.4 que define el nivel físico y el nivel de enlace (capa MAC). Los módulos XBee añaden ciertas funcionalidades a los aportados por el estándar, tales como:

- Detección de paquetes duplicados: Eta funcionalidad no está establecido en la normal y se añade por los módulos XBee.
- Descubrimiento de nodos: cierta información ha sido añadido a la cabecera del paquete para que puedan descubrir otros nodos en la misma red. Se permite un mensaje de descubrimiento de nodos que se enviará, de manera que, el resto de los nodos de red, respondan indicando sus datos.

El envío de los paquetes se pueden hacer sin habilitar y habilitando el cifrado. Se recomienda este último. El cifrado se proporciona a través del algoritmo AES128b. Específicamente a través del tipo AES-CTR. En este caso, el campo Contador de Tramas tiene un identificador único y cifra toda la información contenida en el capo de carga útil, que es el lugar en el marco 802.15.4 donde se almacenan los datos que se enviarán. Se necesitará una llave que se utilizará en el proceso de cifrado. El número de bytes de carga útil (payload) dependerá tanto del modo de transmisión que realiza el paquete y si se envía este en modo cifrado o no cifrado.

El envío de datos es un proceso complejo que necesita algunas estructuras y funciones especiales para llevarse a cabo. Debido a la limitación de las cargas útiles máximas puede ocurrir que el paquete tenga que ser truncado a la longitud máxima posible.

La topología que se utiliza es la siguiente:

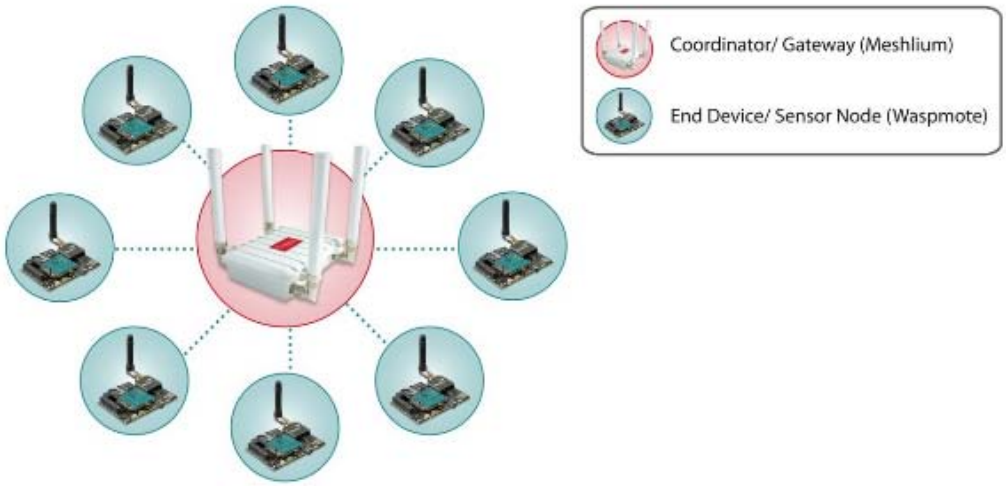


Fig. B.14. Topología en estrella [27]

Como se ve en la imagen, la topología que se sigue es en estrella. Es decir, se necesita un coordinador o nodo central (colocado en el Gateway) que es el responsable de formar una red, la entrega de direcciones y la gestión de otras funciones que definen la red. Está vinculado a todos los demás nodos de a red. El nodo central reúne todos los datos procedentes de los nodos de la red; un router, colocado en el Waspmote y que funciona como dispositivo final o nodo, quien está encargado de recibir y re-direccionar los paquetes de datos a otros dispositivos. Los dispositivos finales pueden recibir y enviar información. Cada aclarar que, el Router, siempre debe estar disponible en la red para recibir los datos, ya que juega un papel importante en la formación de redes, en cambio el dispositivo final, puede estar en un estado de suspensión mientras espera recibir algún dato.

Para poder configurar estos módulos XBee, se requieren una serie de parámetros específicos llamados comandos AT, que permiten establecer una comunicación entre dos módulos. Se necesitará algunos parámetros como:

- Dirección MAC: Necesario para establecer conexiones punto a punto con los nodos hermanos. Cada módulo tiene una dirección MAC única de 64 bits. Permite identificar de forma única un nodo dentro de una red debido a que no puede ser modificado y está dado por el fabricante. Se utiliza en las transmisiones de unidifusión de 64 bits.
- Dirección de Red: Identifica un nodo dentro de una red. La dirección es de 16 bits. Se usa para enviar datos a un nodo en transmisiones de unidifusión de 16 bits.
- PAN ID: número de 16 bits que identifica la red. Debe ser único para diferenciar una red. Todos los nodos en la misma red deberán tener el mismo PAN ID.
- Identificador de nodo: Es una cadena ASCII de 20 caracteres como máximo que identifica el nodo en una red. Se utiliza para identificar un nodo en el nivel de aplicación. También se utiliza para buscar un nodo utilizando su NI.
- Canal: Este parámetro define el canal de frecuencia usado por el módulo para transmitir y recibir. Para que estén en la misma red, todos los nodos deberán estar en el mismo canal.

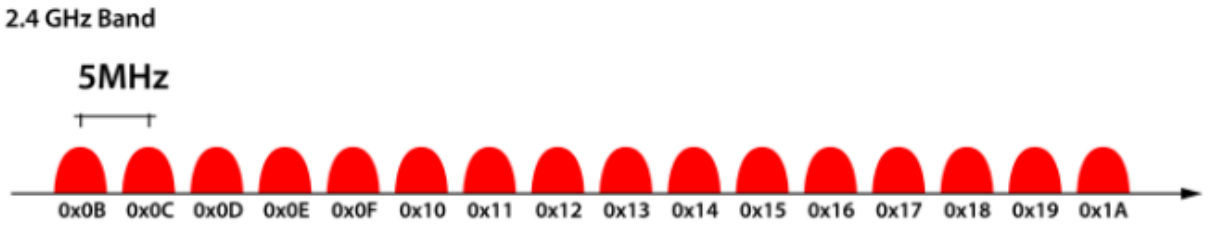


Fig. B.15. Canales de frecuencia en la banda 2.4 GHz [27]

Cada paquete de datos de RF enviado, contiene una dirección de origen y el campo de dirección de destino en su encabezado. Hay dos modos de transmisión en el envío del paquete:

- Unicast (Unidifusión): Es la única que soporta reintentos. Solo se enviaran los paquetes de datos a un único nodo en la red. Mientras esté en este modo, los módulos que reciben, envían una confirmación de recepción de paquetes de RF al transmisor. Es como un modo de aviso de la llegada del paquete al módulo receptor. Direcciones de 16 y 64 bits son compatibles. Con 64 bits, el transmisor debe establecer la dirección MAC del receptor en el paquete transmitido y la dirección de red del receptor debe estar ajustado a 0xFFFF.
- Broadcast (Multidifusión): Se utiliza para enviar el paquete a todos los nodos en una red. Para enviar un mensaje de difusión, la dirección de destino de 64 bits se debe establecer en 0x000000000000FFFF. Este modo no devuelve ACK por lo tanto, no se puede confirmar si el paquete se ha recibido correctamente.

B.3.2. XBee 868

El Módulos XBee-PRO 868 [28] proporcionan una conectividad inalámbrica de largo

alcance a los dispositivos finales. Estos módulos utilizan un protocolo patentado punto a multipunto 868 MHz para aplicaciones europeas.

Fig. B.16. Modulo 868 con la antena de 4.5dBi [28]

Estos módulos son ideales para entornos difíciles donde la penetración de RF y la distancia de transmisión son críticos para la aplicación.

Como parte de la familia XBee de Digi de productos de RF, estos módulos son fáciles de usar.

Disponen de un solo canal de trabajo:

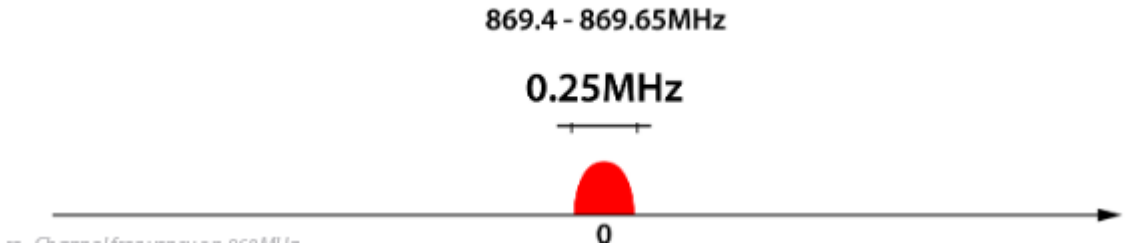


Fig. B.17. Frecuencia del canal [28]

En este módulo, al trabajar en un único canal, solo hace falta configurar el PAN ID para que los dos módulos trabajen en la misma red.

B.3.3. XBee DigiMesh

Los Módulos DigiMesh [29] utilizan el protocolo DigiMesh punto por punto en 2.4 GHz para despliegues globales. Este protocolo de malla innovador ofrece a los usuarios añadir estabilidad a la red a través de la auto-sanación, auto-descubrimiento y operación de la red densa. DigiMesh es ideal para aplicaciones sensibles de energía dependiendo de baterías o tecnología de recolección de energía por energía.

Los módulos 802.15.14 y 900 pueden usar un microprograma opcional (DigiMesh) de manera que son capaces de crear redes de mallas en lugar de la topología punto a punto usual. Este microprograma ha sido desarrollado por Digi para permitir a los módulos dormirse, sincronizarse y trabajar en igualdad de condiciones, evitando el uso de nodos de router o coordinadores que tienen que estar permanentemente encendido.

Las características del protocolo implementado son:

- Auto curación: cualquier nodo puede unirse o abandonar la red en cualquier momento.
- Todos los nodos son iguales: No existen relaciones entre padres e hijos.
- Descubrimiento de ruta: en lugar de mantener un mapa de la ruta, estas se descubren cuando ellas son necesitadas.
- ACK´S selectivos: solo el destinatario responde para enrutar los mensajes.
- Fiabilidad: el uso de ACK´S garantiza la fiabilidad de transmisión de datos.

Los módulos XBee DigiMesh comparten el módulo de hardware con el XBee-802.15.4 y 900, siendo capaz de pasar de una a otra cambiando el firmware. Por esta razón, las características relacionadas con el hardware son los mismos, cambiándolo en relación al protocolo utilizado.

El proceso de cambiar el firmware se puede hacer con un programa X-CTU de Digi y con un Gateway.

En este proyecto, se ha cambiado el firmware Xbee de 802.15.4 a DigiMesh, con lo cual la banda de frencuencias es la misma que la del 802.15.4.

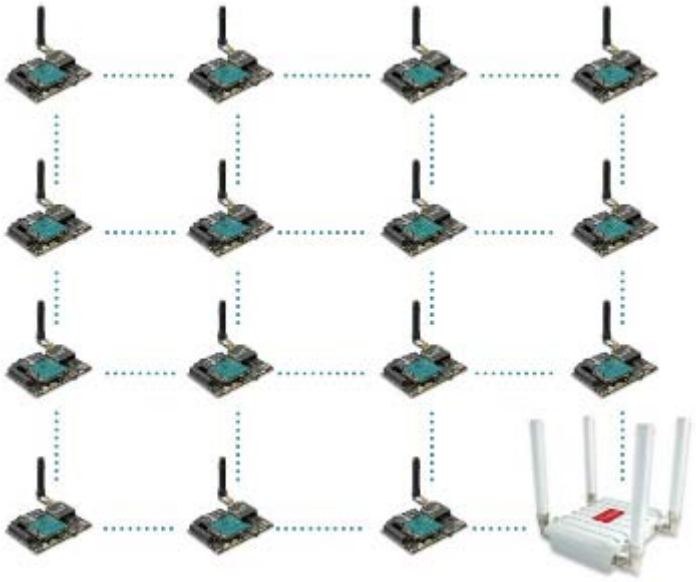


Fig. B.18. Topología de red [29]

B.3.4. XBee ZigBee

Este módulo [30] está basado en el estándar 802.15.4. ZigBee hace referencia al protocolo de comunicación empleado para la comunicación.

Fig. B.19. Módulo ZigBee [30]

ZigBee define tres tipos diferentes de dispositivos: el coordinador, el router y los dispositivos finales. A continuación, se muestra un ejemplo de una posible red:

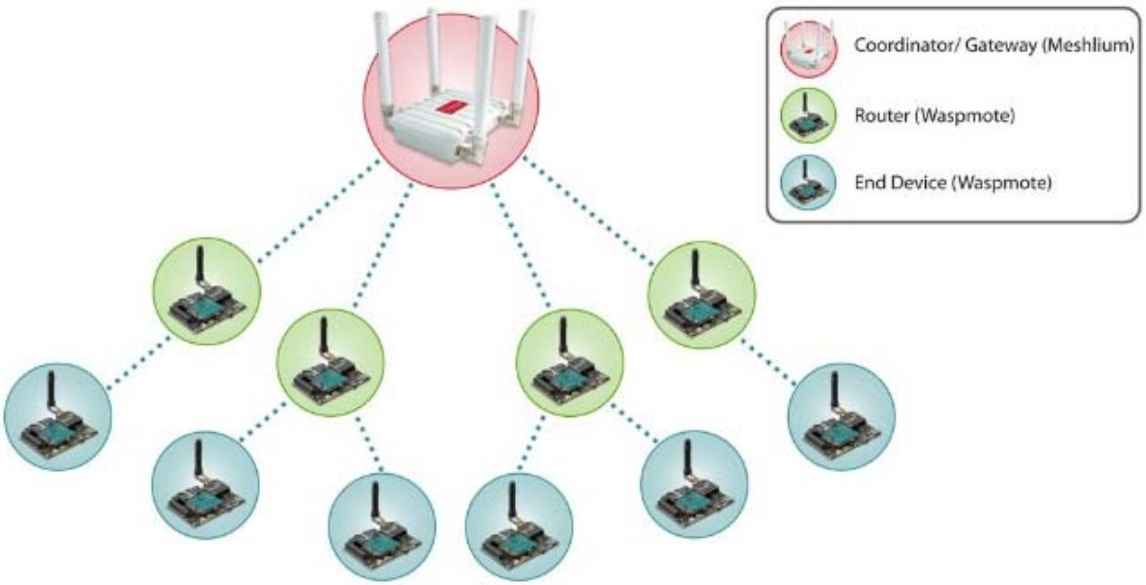


Fig. B.20. Topología en árbol [30]

Coordinador:

Cada red ZigBee debe tener un coordinador. Tiene las siguientes características:

- Selecciona el canal y el PAN ID (tanto 64 bits como 16 bits) para comenzar la red.
- Permite que los routers y los dispositivos finales se unan a la red.
- Ayudan en el encaminamiento de los datos.
- No puede dormirse. Siempre tiene que estar despierto.

Router:

Un router tiene las siguientes características:

- Debe unir una red ZigBee antes de que pueda transmitir, recibir o rutear datos.
- Después de la conexión, permite que los routers y los dispositivos finales se unan a la red.
- Después de la conexión, puede rutear datos.
- No puede dormirse. Siempre tiene que estar despierto.

Dispositivo final:

Un dispositivo final tiene las siguientes características:

- Debe unir una red ZigBee antes de que pueda recibir o transmitir datos.
- No permite que los dispositivos se unan a la red.
- Debe siempre transmitir y recibir datos de RF.
- No puede rutear datos.
- Puede dormirse.

En las redes ZigBee, el coordinador debe seleccionar un PAN ID (64 bits y 16 bits) y el canal para comenzar una red. Después de eso, se comporta esencialmente como un router. El coordinador y los routers permiten que otros dispositivos se unan a la red y rutear datos.

Después de un dispositivo final una un router o coordinador, debe ser capaz de transmitir o recibir datos de RF a través de ese el router o coordinador. El router o coordinador que permita que el dispositivo final se una llegará a ser su padre. Dado que el dispositivo final puede dormir, el padre debe ser capaz de amortiguar o conservar los paquetes de datos entrantes destinados al dispositivo final, hasta que pueda despertarse y reciba los datos.

B.3.5. Wifi

El módulo Wi-Fi [31] para la plataforma Waspmote completa las posibilidades de conectividad actuales permitiendo la comunicación directa de los nodos de sensores con cualquier

router WiFi en el mercado. Además de esto, esta radio permite que Waspmote envie directamente la información a cualquier iPhone o Android Smartphones y sin la necesidad de un router intermedio, que hace posible crear redes de sensores WiFi en cualquier lugar utilizando sólo Waspmote y un dispositivo móvil corriendo con baterías.

Con esta radio, Waspmote puede realizar conexiones HTTP recuperarando y enviando información a la web y FTP en ambos modos normales y seguras (HTTPS / FTPS), así como el uso de TCP / IP y sockets UDP / IP para conectarse a cualquier servidor situado en la Internet.

Fig. B.21. Módulo Wifi [31]

B.3.6. Bluetooth

El módulo bluetooth [32] ha sido diseñado principalmente para descubrir hasta 250 dispositivos en una zona variable. El módulo pertenece a la solución de Smart Cities creada por Libelium, permitiendo aplicaciones como el control del tráfico de vehículos y peatones.

Por otra parte, una API ha sido creada para gestionar las consultas de los module. Esta API está diseñada también para propósitos de descubrimiento de dispositivos y los intercambios de datos básicos, dejando para el futuro otras aplicaciones como el intercambio de datos complejos en una red Bluetooth.

Tiene que ser mencionado que los procesos de consulta de módulo bluetooth son anónimas debido a sólo la dirección MAC se obtiene del Bluetooth del dispositivo remoto. No se obtienen ninguna cuenta ni números de teléfono. Este hecho permite guardar la privacidad de los usuarios del bluetooth.

Hay algunos parámetros que pueden ser consultados desde dispositivos dentro del área de

detección. Los más importantes se describen a continuación:

- Dirección MAC: Es el número de identificación único del dispositivo Bluetooth. Cuenta con 12 dígitos hexadecimales.
- CoD (Clase de dispositivo): los dispositivos Bluetooth se clasifican de acuerdo con el dispositivo que se integran. Por lo tanto, un dispositivo de manos libres del vehículo pertenecerá a una clase diferente que un teléfono móvil de peatones. Este parámetro tiene 6 dígitos hexadecimales y permite distinguir si el dispositivo Bluetooth detectado es un vehículo, un peatón y así sucesivamente.
- RSSI: Este parámetro muestra la calidad de la señal recibida, es decir, con que fuerza llega. Se puede utilizar para conocer la distancia entre el módulo Bluetooth y el dispositivo analizado. Se muestra como un valor negativo entre -40 dBm (dispositivos cercanos) y -90 dBm (dispositivos de lejos).

Además, hay otro parámetro que puede ser consultado desde dispositivos Bluetooth. Este parámetro se llama "*Friendly name*" y se define por el propietario del dispositivo bluetooth. Es sólo una forma "amigable" para nombrar un dispositivo bluetooth dentro de la dirección MAC.

Fig. B.22. Dispositivo Bluetooth [32]

B.3.7. BLE

El módulo Bluetooth Low Energy [33] es compatible con el nuevo Bluetooth 4.0. estándar.

El estándar Bluetooth 4.0, también conocida como Bluetooth Low Energy (BLE), es una nueva tecnología de radio de corto alcance, optimizado para aplicaciones de baja potencia. Es diferente del clásico Bluetooth (BR / EDR), pero con los mismos beneficios como la robustez, la interoperabilidad, libre derecho o la conectividad con los smarth phones y PCs.

Sin embargo, la compatibilidad con los dispositivos Bluetooth clásicos depende de la marca dispositivo.

Por otra parte, se ha creado una API para gestionar el módulo, lo que permite configurar las características principales, representar análisis o conectar con otros dispositivos.

El módulo BLE es administrado por UART y puede ser conectado en los SOCKET0 y SOCKET1 de Waspmote.

Fig. B.23. Dispositivo BLE [33]

B.3.8. GSM/GPRS

Waspmote puede integrar un módulo GSM (Sistema Global para Comunicaciones Móviles) y GPRS (servicio general de paquetes vía radio) para permitir la comunicación utilizando la red de telefonía móvil.

El módulo GPRS [34] o servicio general de paquetes vía radio es una extensión del Sistema Global para Comunicaciones Móviles para la trasmisión de datos mediante conmutación de paquetes.

Con GPRS, se puede utilizar servicios como protocolo de aplicación inalámbrico, servicio de mensajes cortos, servicio d mensajería multimedia, Internet y para los servicios de comunicación, como el correo electrónico, etc. Para fijar una conexión GPRS para un módem inalámbrico, el usuario debe especificar un APN (nombre del punto de acceso), opcionalmente un nombre y contraseña de usuario y muy raramente una dirección IP, todo proporcionado por el operador d red.

Con esta radio, Waspmote puede realizar conexiones HTTP recuperando y enviando información a la web y FTP en ambos modos normales y seguras (HTTPS / FTPS), así como el uso de TCP / IP y sockets UDP / IP para conectarse a cualquier servidor situado en la Internet.

Este módulo requiere de la colocación de una tarjeta SIM a la que se le pondrá su pin correspondiente.

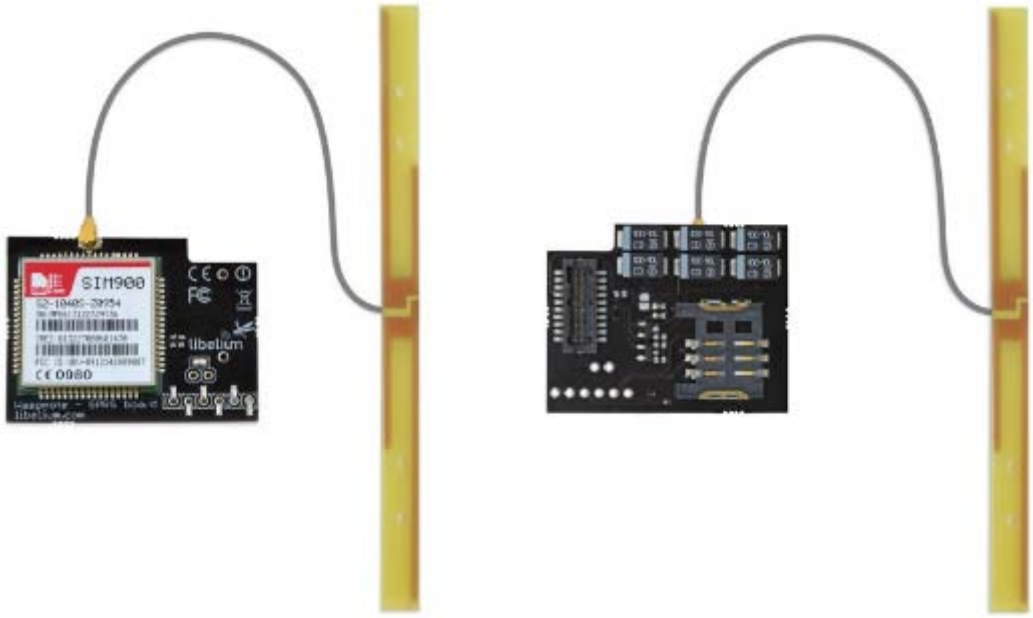


Fig. B.24. Cara superior y posterior del módulo GSM/GPRS con su antena [34]

B.3.9. 3G+GPS

Waspmote puede integrar un módulo UMTS (Sistema de Telecomunicaciones Móviles Universales basados en la tecnología WCDMA) y GPRS (Servicio general de paquetes vía radio) para permitir la comunicación utilizando la red de telefonía móvil 3G / GPRS.

Este módulo también cuenta con un GPS interno lo que permite la localización del dispositivo al aire libre y en el interior combinando tramas NMEA estándar con triangulación de teléfonos móvil ID usando modos A-GPS y S-GPS.

El módulo de comunicación 3G [35] está orientado especialmente para trabajar con servidores de Internet implementando internamente varios protocolos de capa de aplicación que hacen más fácil enviar la información a la nube. Podemos hacer la navegación HTTP y HTTPS (en modo seguro), carga y descarga de contenido a un servidor web. De la misma manera, los protocolos FTP y FTPS también están disponibles para subir archivos. El usuario puede incluso enviar y recibir correos electrónicos directamente desde Waspmote usando clientes SMTP implementado internamente.

Este módulo junto con la placa de Video Cámara, explicada en el apartado anterior, es capaz de tomar fotos y grabar videos y subirlos en tiempo real a la nube.

El módulo 3G viene con una tarjeta SD interna de 2 GB (ampliado hasta 32 GB) que se utiliza para almacenar las fotos y videos tomados por la cámara y para subir archivos creados a FTP desde el propio módulo.

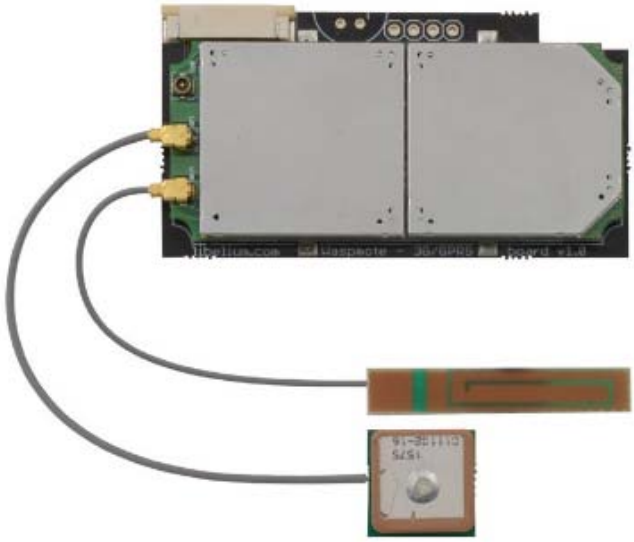


Fig. B.25. Módulo 3G con sus respectivas antenas para el mismo y para el GPS [35]

B.3.10. GPS

Waspmote puede integrar un receptor GPS [36] que permite conocer la ubicación exacta del mote en cualquier momento. De esta manera, se puede obtener la posición exacta del mote e incluso la hora y fecha actual, para sincronizar el reloj interno de Waspmote (RTC) con el tiempo real.

Debido a que la señal GPS es débil dentro de edificios, se recomienda utilizar el módulo GPS al aire libre, con una línea de visión directa hacia el cielo. Esto asegurará la calidad de la señal necesaria para obtener datos válidos de GPS.

El módulo GPS nos da información sobre la latitud, longitud, altitud, dirección, velocidad y fecha/hora.

Fig. B.26. Módulo GPS [36]

Anexo C: Desarrollo de las pruebas realizadas junto con el cálculo del consumo total de cada proceso.

A continuación, se va a exponer y explicar el resto de las pruebas de los tres grupos desarrollados (funcionalidades, sensores y módulos de comunicación), indicando el dispositivo medido, breve explicación de lo que se ha medido, las gráficas correspondientes como resultado del código ejecutado, una tabla del consumo de cada proceso ejecutado y una serie de comentarios acerca de la medida realizada (solo en los casos que sean necesarios).

Comentar que, el código ejecutado de cada proceso, se encuentra en la carpeta 3_SW en el CD, permitiéndose abrirlos en formato Word si no disponen del programa utilizado.

C.1. Funcionalidades de la plataforma Waspmote.

C.1.1. Proceso On y Off de los Leds

C.1.1.1. Led 0

En este ejemplo se muestra el proceso On y Off del Led 0 de Waspmote.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\lED0_on_off

Se muestra las gráficas como resultado del ejemplo ejecutado. Se observa en la gráfica de la izquierda la medida general del consumo de la puesta a On y Off del led y en la de la derecha, el tiempo total de ejecución del proceso completo, que, como se ve, se corresponde con el impuesto en el código.

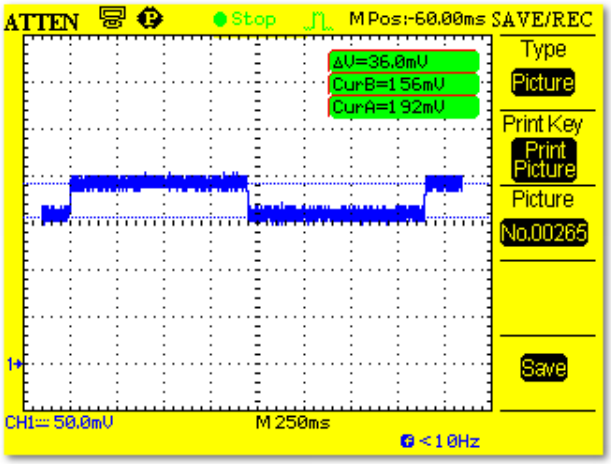


Fig. C.1.a. Consumo Led 0

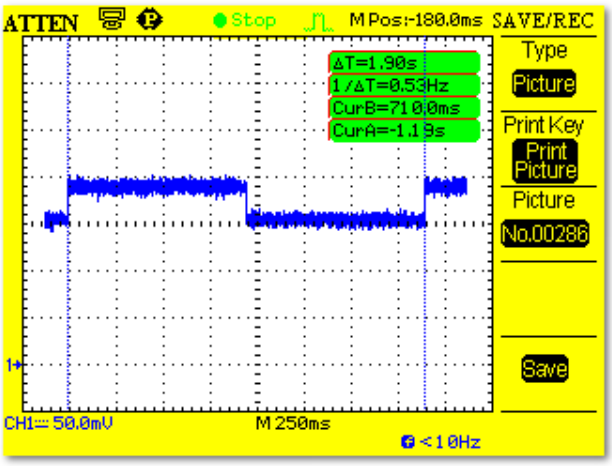


Fig. C.1.b. Tiempo ejecución Led 0

En la siguiente tabla, se muestra el consumo del proceso On, estado On y proceso Off del Led:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso On	0	3.6	0
Estado On	1000	3.6	3600
Proceso Off	0	3.6	0

Tabla C.1. Consumo del proceso y estado on y proceso off del Led 0

Se observa que ejecutar las funciones del proceso a On y del proceso a Off son instantáneas.

C.1.1.2. Led 1

En este ejemplo se muestra el proceso On y Off del Led 1 de Waspmote.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\lED1_on_off

Se muestra las gráficas como resultado del ejemplo ejecutado. Se observa en la gráfica de la izquierda la medida general del consumo de la puesta a On y Off del led y en la de la derecha, el tiempo total de ejecución del proceso completo, que, como se ve, se corresponde con el impuesto en el código.

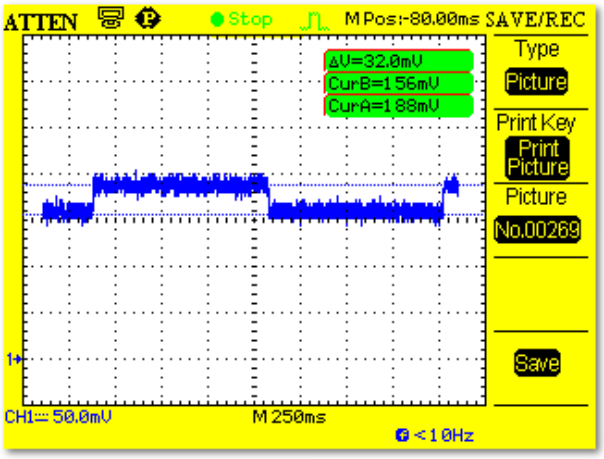


Fig. C.2.a. Consumo Led 1

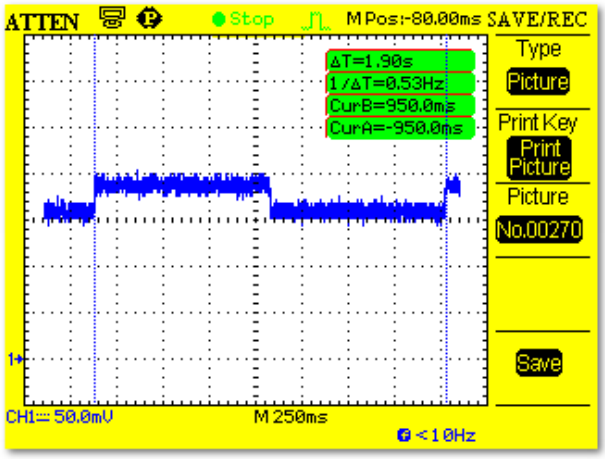


Fig. C.2.b. Tiempo ejecución Led 1

En la siguiente tabla, se muestra el consumo del proceso On, estado On y proceso Off del Led:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso On	0	3.2	0
Estado On	1000	3.2	3200
Proceso Off	0	3.2	0

Tabla C.2. Consumo del proceso y estado on y proceso off del Led 1

Se observa que ejecutar las funciones del proceso a On y del proceso a Off son instantáneas.

C.1.2. Memoria EEPROM.

C.1.2.1. Lectura EEPROM

En este ejemplo, se muestra cómo se lee en la memoria EEPROM a partir de la dirección 1024.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\eeprom_read

Se muestra la gráfica como resultado del ejemplo ejecutado:

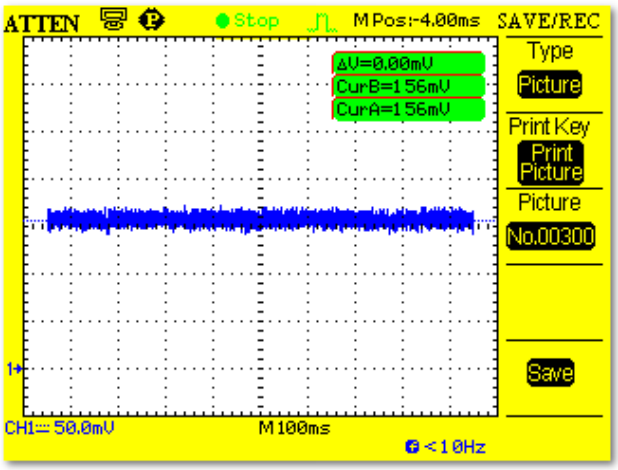


Fig. C.3. Lectura de la EEPROM

Se observa que la gráfica sale totalmente plana ya que ejecutar la función de lectura de la EEPROM es instantánea, además de que el consumo es totalmente nulo.

Por lo tanto:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Read EEPROM	0	0	0

Tabla C.3. Consumo de la lectura de la EEPROM

C.1.2.2. Escritura EEPROM

En este ejemplo, se muestra cómo se escribe en la memoria EEPROM a partir de la dirección 1024.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\eeprom_write

Se muestra la gráfica como resultado del ejemplo ejecutado:

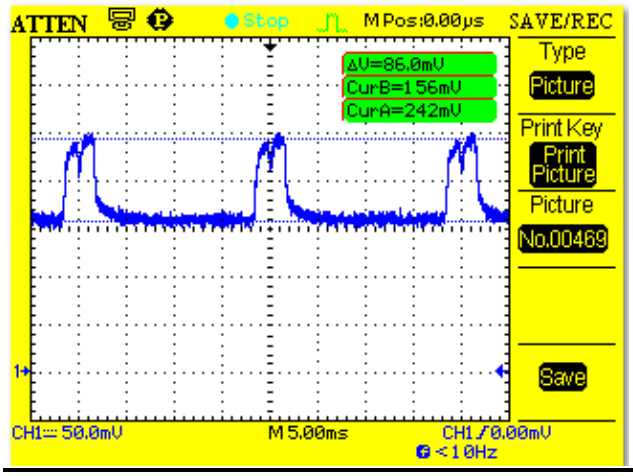


Fig. C.4.a. Lectura EEPROM cada 20 ms

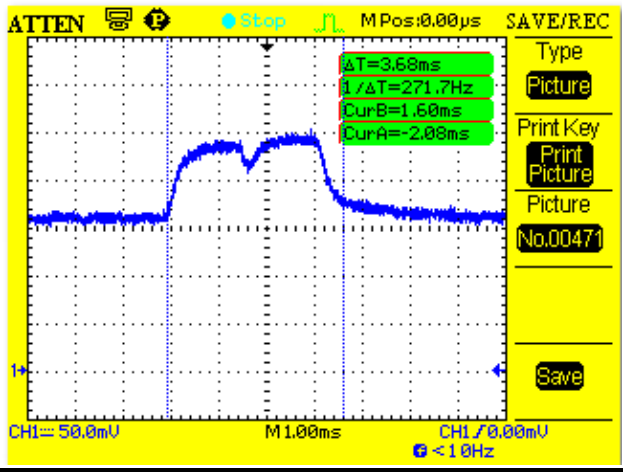


Fig. C.4.b. Tiempo ejecución lectura EEPROM

En la siguiente tabla, se muestra el consumo de la escritura de la EEPROM del Waspmote:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Escritura EEPROM	3.7	7	25.9

Tabla C.4. Consumo de la escritura de la EEPROM

C.1.2.3. Escritura y lectura de la EEPROM

En este ejemplo, se muestra cómo se escribe y posteriormente, se lee en la memoria EEPROM a partir de la dirección 1024.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\eeprom_write_read

Se muestra la gráfica como resultado del ejemplo ejecutado:

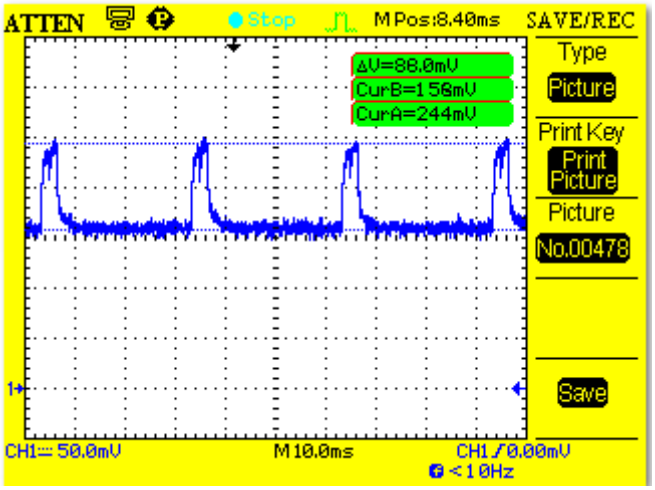


Fig. C.5. Escritura y lectura de la EEPROM

Se observa que solamente consume la escritura de la EEPROM, como era de esperar ya que la lectura no consume nada. Por lo tanto, el consumo es el mismo, es decir:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Escritura/lectura de la EEPROM	3.7	7	25.9

Tabla C.5. Consumo total de la escritura y lectura de la EEPROM

C.1.3.1. RTC.

C.1.3.1.1. Proceso On y Off

En este ejemplo, se muestra el proceso On, estado On y proceso Off del RTC del Waspmote.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\rtc_on_off

Se muestra las gráficas como resultado del ejemplo ejecutado:

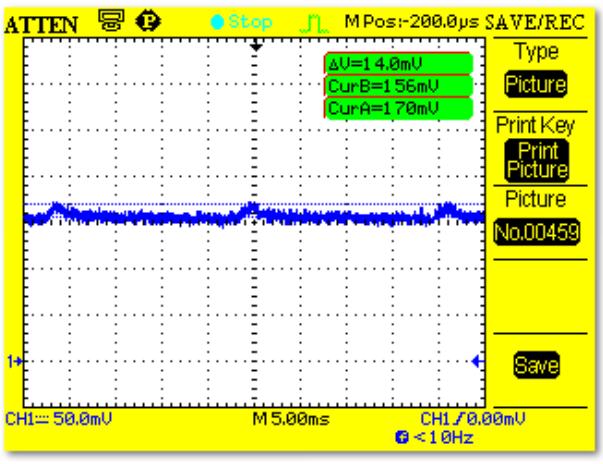


Fig. C.6.a. Medida general del proceso On y Off del RTC

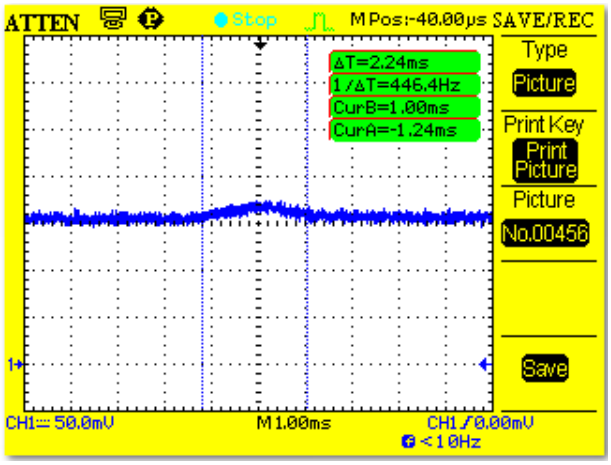


Fig. C.6.b. Tiempo de ejecución del proceso On

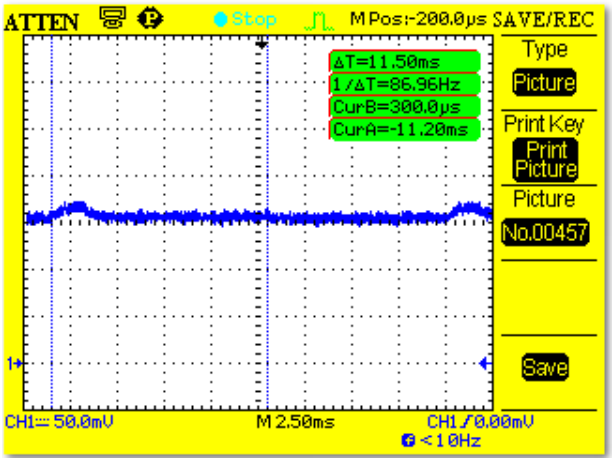


Fig. C.6.c. Tiempo de ejecución del proceso y estado On

En la siguiente tabla, se muestra el consumo del proceso On, estado On y proceso Off del RTC:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	2.2	1.3	2.86
Estado On	10	0.4	4
Proceso Off	0	0.4	0

Tabla C.6. Consumo del proceso y estado on y estado off del RTC

Como se ve, el tiempo en el estado On corresponde con el tiempo que se ha puesto en el código, es decir los 10 ms. Cada destacar que el tiempo de ejecución del proceso a Off es nulo.

Se optó por poner un delay muy pequeño entre estado y estado para poder distinguir bien los procesos ya que el tiempo de ejecución y el consumo es muy bajo. Pero siguiendo el mismo proceso para todos, se toma como referencia tanto para el estado On como el Off de todos los estados "indefinidos" una duración de 1000 ms para que todos duren igual y sean exactos. Con lo cual, el consumo en el estado On sería:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Estado On	1000	0.4	400

Tabla C.7. Consumo del estado on con 1000 ms del RTC

C.1.3.2. Fijar fecha y hora

En este ejemplo se muestra como poner la fecha y hora utilizando el RTC del Waspmote.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\rtc_on_setting

Se muestra las gráficas como resultado del ejemplo ejecutado:

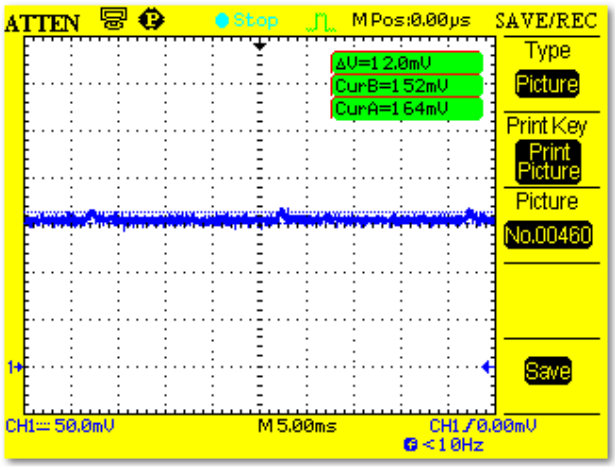


Fig. C.7.a. Medida general cada 20 ms

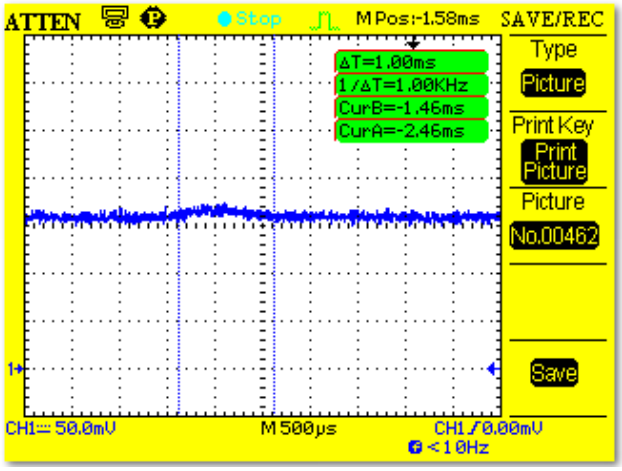


Fig. C.7.b. Tiempo de ejecución de la función

En la siguiente tabla se muestra el consumo del proceso. Para ello, se ha dejado encendido el RTC en el *setup* y así diferenciar el consumo de cada proceso y observar con facilidad lo que consume en concreto este ejemplo, que es lo que se busca:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Fijar fecha/hora	1	0.8	0.8

Tabla C.8. Consumo de poner fecha y hora en el RTC

C.1.3.3. Leer fecha y hora

En este ejemplo se muestra como leer la fecha y hora utilizando el RTC del Waspmote.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\rtc_on_getting

Se muestra las gráficas como resultado del ejemplo ejecutado:

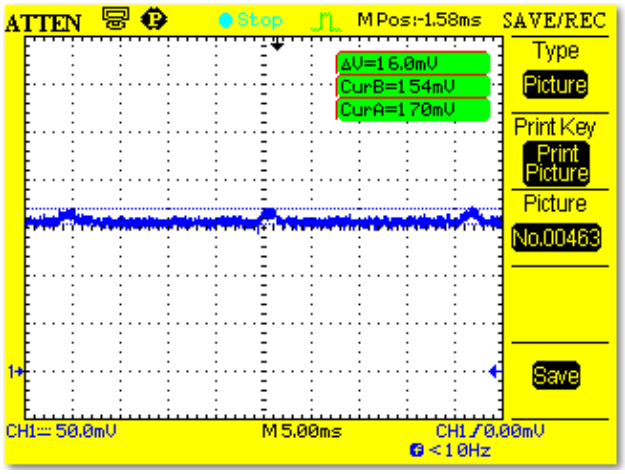


Fig. C.8.a. Medida general cada 20 ms

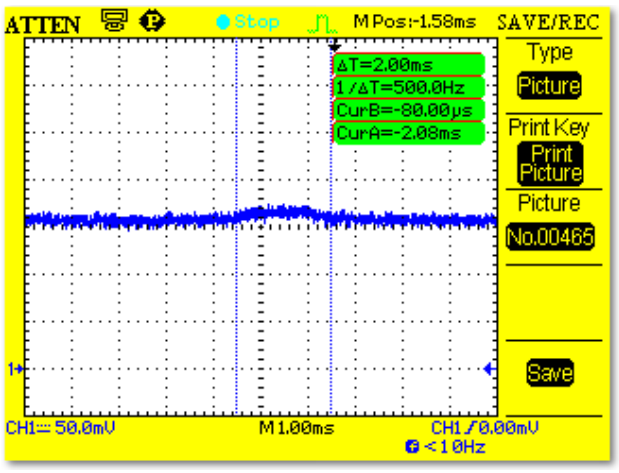


Fig. C.8.b. Tiempo de ejecución de la función

En la siguiente tabla se muestra el consumo del proceso. Para ello, se ha dejado encendido el RTC en el *setup* y así diferenciar cada proceso y observar con facilidad lo que consume leer la fecha y hora, que es lo que se busca:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Leer fecha/hora	2	1.2	2.4

Tabla C.9. Consumo de leer fecha y hora en el RTC

C.1.3.4. Leer temperatura

En este ejemplo se muestra como leer la temperatura del RTC del Waspmote, usando su sensor de temperatura.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\rtc_temperature

Se muestra las gráficas como resultado del ejemplo ejecutado:

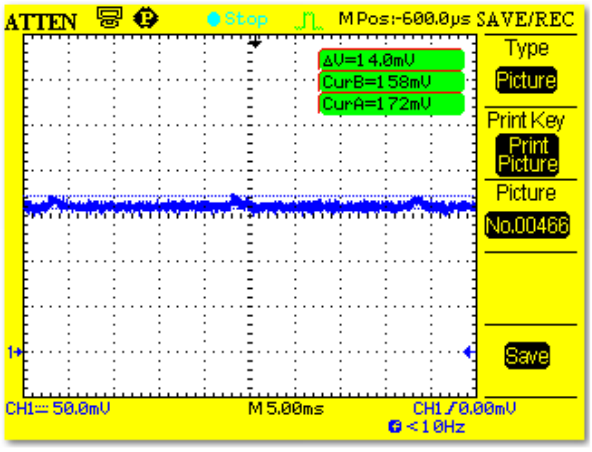


Fig. C.9.a. Medida general cada 20 ms

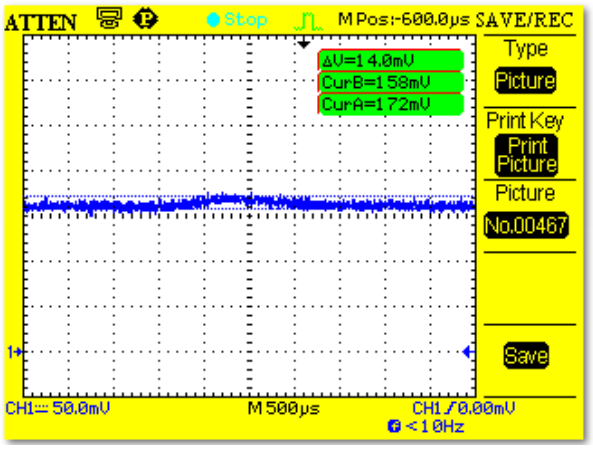


Fig. C.9.b. Zoom del proceso de lectura de la Tª

En la siguiente tabla se muestra el consumo del proceso. Para ello, se ha dejado encendido el RTC en el *setup* y así diferenciar cada proceso y observar con facilidad lo que consume leer la temperatura, que es lo que se busca:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Leer temperatura	2	1.2	2.4

Tabla C.10. Consumo de la lectura de temperatura en el RTC

C.1.4. Sistemas de Energía.

C.1.4.1. Modo Deep Sleep

En este ejemplo se muestra como poner Waspmote en un modo de consumo de baja energía, despertándose usando el RTC. En el código, se ha optado por despertarse al cabo de 1 sg.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\mode_deep_sleep

Se muestra las gráficas como resultado del ejemplo ejecutado:

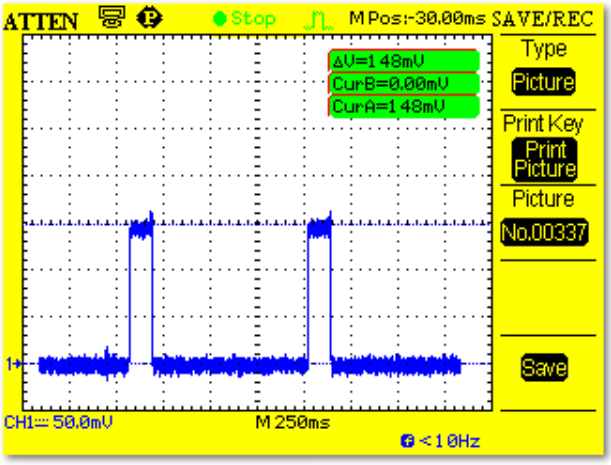


Fig. C.10.a. Medida general

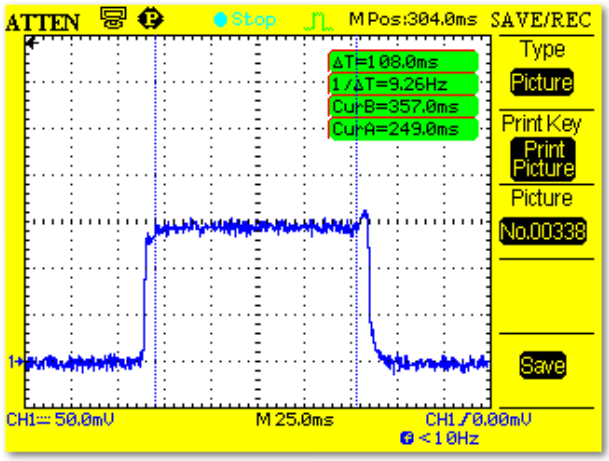


Fig. C.10.b. El despierte y el echarse a dormir detallado

Para saber cuánto tiempo le cuesta despertarse y echarse a dormir, se ha utilizado una serie de funciones (que están comentadas en el código) que lo que hacen es poner el pin digital 8 en alto y en bajo y así poder distinguir y separar facilmente el tiempo que le cuesta despertarse el Waspmote y echarse a dormir:

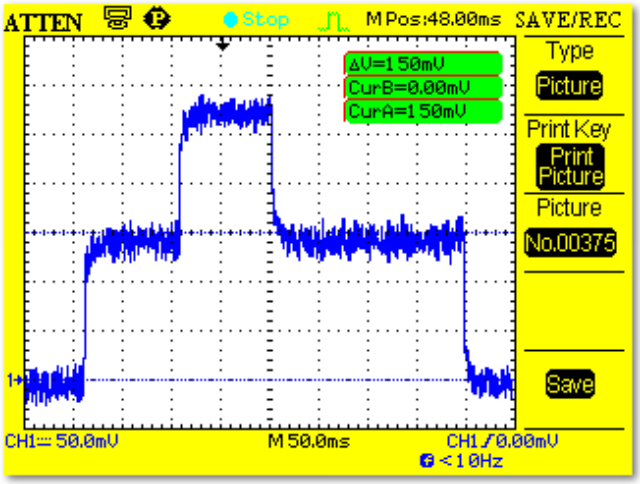


Fig. C.10.c. Poniendo el pin digital en alto y bajo

Con esta gráfica, se comprueba que hay una gran diferencia entre el tiempo que le cuesta despertarse Waspmote, que es muy poco, y el que le cuesta echarse a dormir.

En la siguiente tabla se muestra el consumo del proceso:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Despertarse	8	9.55	76.4
Echarse a dormir	115	15.6	1794
Durmiendo	877	0.055	48.23

Tabla C.11. Consumo de los tres procesos en el modo Deep-Sleep

C.1.4.2. Nivel de Batería

En este ejemplo muestra cómo obtener el nivel de batería restante.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\nivel_bateria

Se muestra las gráficas como resultado del ejemplo ejecutado:

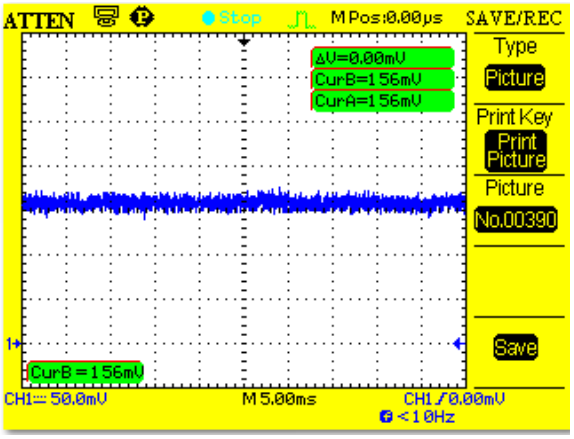


Fig. C.11. Lectura del nivel de batería

Se observa que la gráfica sale continua y constante. Esto es debido a que leer el nivel de batería hace que no se produzca ningún tipo de consumo. Sin embargo, ejecutar la función de la lectura del nivel de batería es muy bajo:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Leer nivel de Batería	1	0	0

Tabla C.12. Consumo de la lectura del nivel de batería

C.1.4.3. Modo Hibernate

Este ejemplo muestra cómo configurar Waspmote en el modo de consumo de energía más bajo (llevarlo a hibernate), desconectando toda la placa salvo el RTC, que se alimenta con la batería auxiliar.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\mode_hibernate

Se muestra las gráficas como resultado del ejemplo ejecutado:

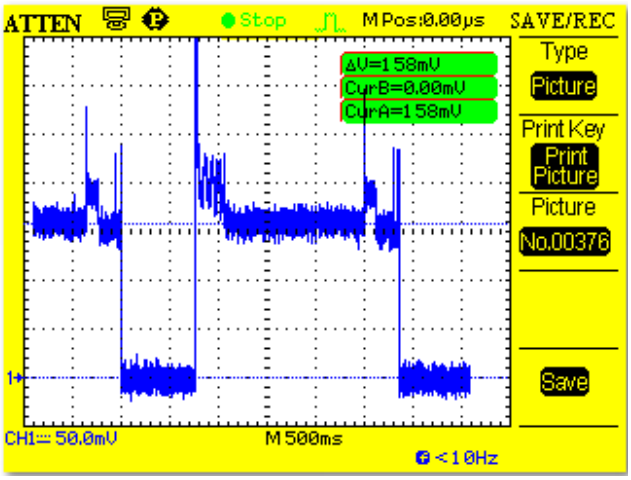


Fig. C.12.a. Medida general del proceso

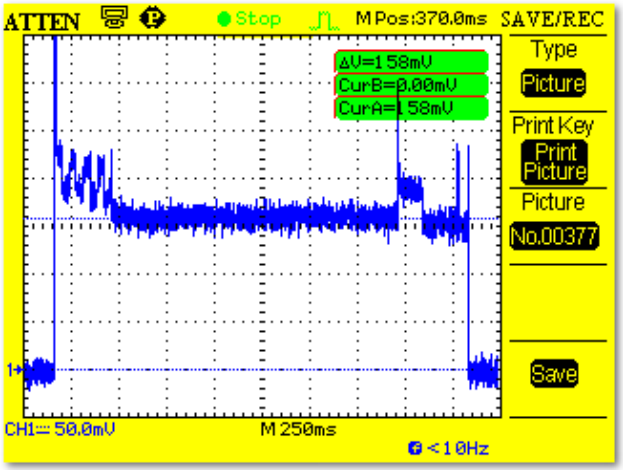


Fig. C.12.b. Zoom de la medida general

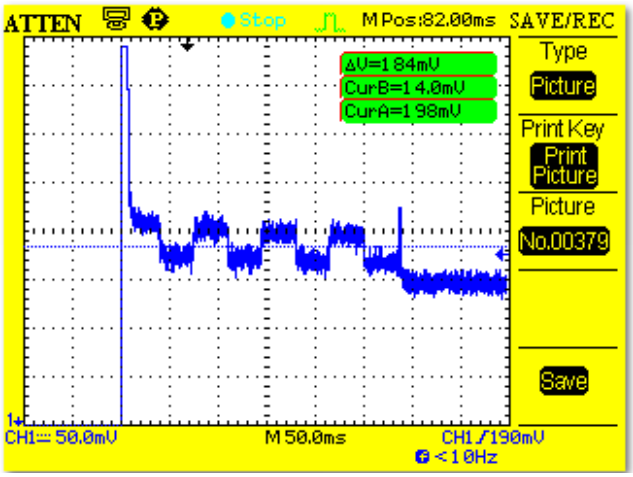


Fig. C.12.c. Zoom de los picos del principio

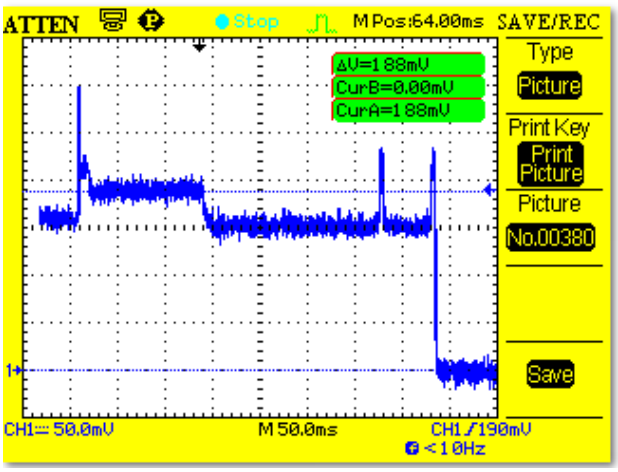


Fig. C.12.d. Proceso de irse a dormir

Aplicando el mismo proceso de Deep Sleep, podemos averiguar hasta donde dura cada proceso que se produce en el modo *hibernate*. Se ve que, aunque se haya puesto un tiempo de

estado en *hibernate* de 2 sg, no está en este estado todo ese tiempo, ya que se produce un tiempo de despierte desde *hibernate*, un tiempo de comprobación de si va a entrar a *hibernate* y posteriormente un tiempo de irse a dormir o irse a *hibernate*, *por lo tanto* el estado en *hibernate* se reduce respecto al tiempo programado.

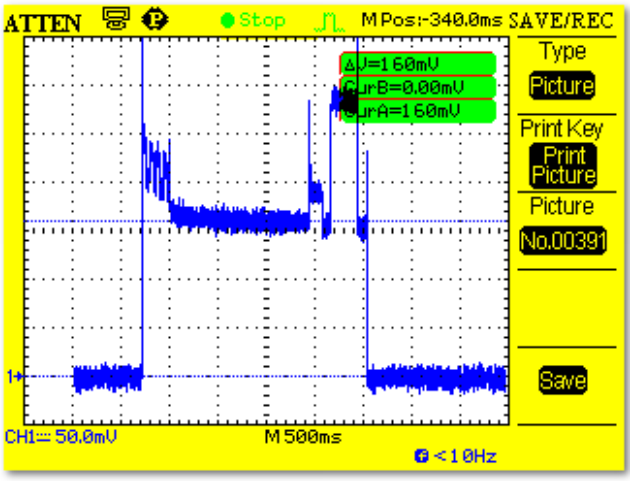


Fig. C.12.e. Poniendo el pin digital en alto y bajo

En la siguiente tabla se muestra el consumo de los diferentes procesos:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Despertarse	1915	16.72	32022
Proceso de entrar a Hibernate	187	15.6	2917.2
Echarse a dormir	60	16.93	1016
Durmiendo	775	0.00006	0.0465

Tabla C.13. Consumo de los tres procesos en el modo Hibernate

C.1.5. Acelerómetro.

C.1.5.1. Proceso On y Off

En este ejemplo se muestra el proceso On, estado On y proceso Off del Acelerómetro de Waspmote.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\acceleration_on_off

Se muestra las gráficas como resultado del ejemplo ejecutado:

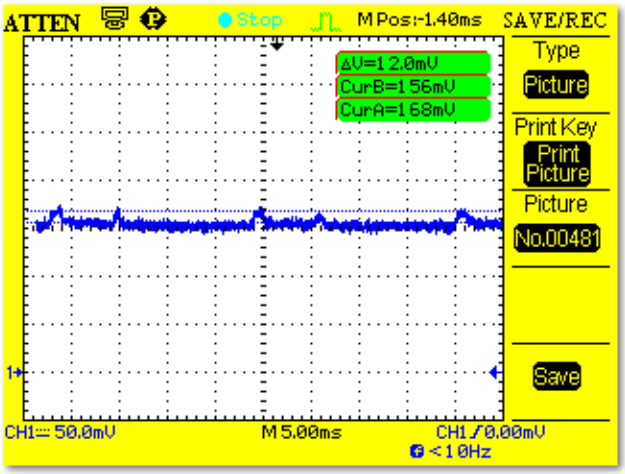


Fig. C.13.a. Medida general

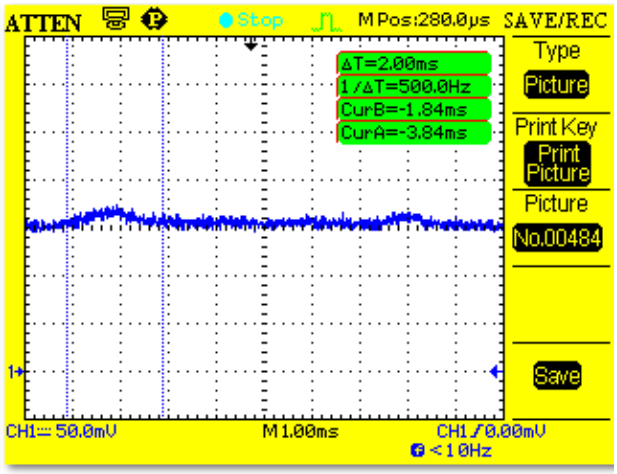


Fig. C.13.b. Tiempo ejecución del proceso a On

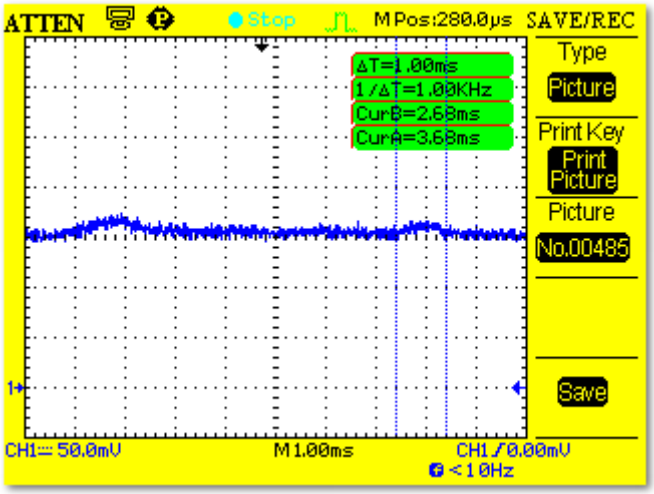


Fig. C.13.c. Tiempo ejecución del proceso a Off

En la siguiente tabla se muestra el consumo de los diferentes procesos, de acuerdo al código que se ha ejecutado:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	2	1.1	2.2
Estado On	5	0.4	20

Proceso Off	1	0.8	0.8

Tabla C.14. Consumo del proceso y estado on y proceso off del acelerómetro

Se optó por poner un delay muy pequeño entre estado y estado para poder distinguir bien los procesos ya que el tiempo de ejecución y el consumo es muy bajo. Pero siguiendo el mismo proceso para todos, se toma como referencia tanto para el estado On como el Off de todos los estados "indefinidos" una duración de 1000 ms para que todos duren igual y sean exactos. Con lo cual, el consumo en el estado On sería:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Estado On	1000	0.4	400

Tabla C.15. Consumo del estado on con 1000 ms del acelerómetro

C.1.5.2. Lectura del Acelerómetro

En este ejemplo se muestra cómo obtener la aceleración en los distintos ejes usando las funciones más básicas relacionadas con el acelerómetro de Waspmote.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\acceleration_reading

Se muestra las gráficas como resultado del ejemplo ejecutado:

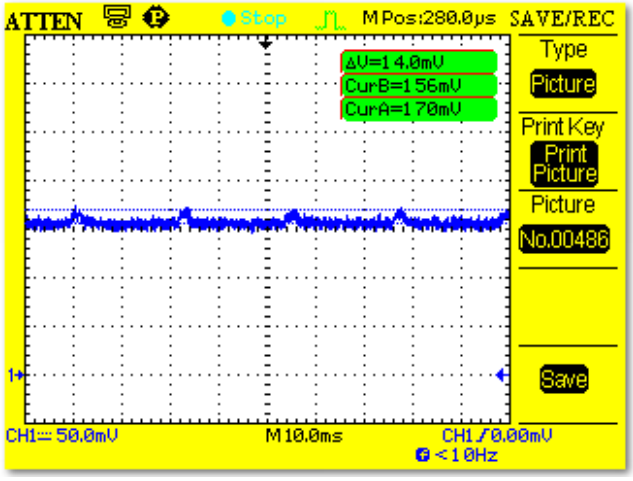


Fig. C.14.a. Medida general cada 20 ms

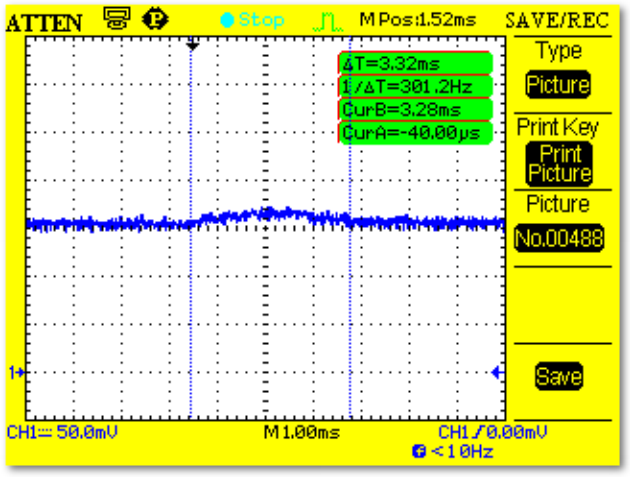


Fig. C.14.b. Tiempo de ejecución de los 3 ejes

En la siguiente tabla se muestra el consumo de la lectura de los ejes del acelerómetro:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Lctura Acelerómetro	3.32	0.96	3.18

Tabla C.16. Consumo de la lectura del acelerómetro

C.1.6. Tarjeta Micro-SD.

C.1.6.1. Proceso On y Off

En este ejemplo se va a medir el proceso On y Off de la tarjeta micro-SD.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\sd_on_off

Se muestra las gráficas como resultado del ejemplo ejecutado:

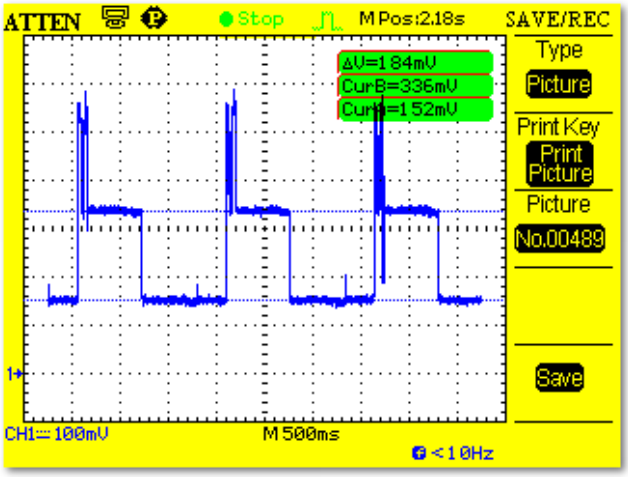


Fig. C.15.a. Medida general cada 500 ms

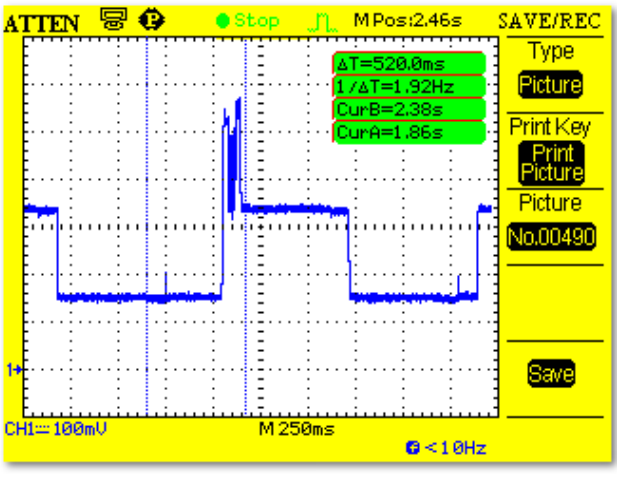


Fig. C.15.b. Proceso a On

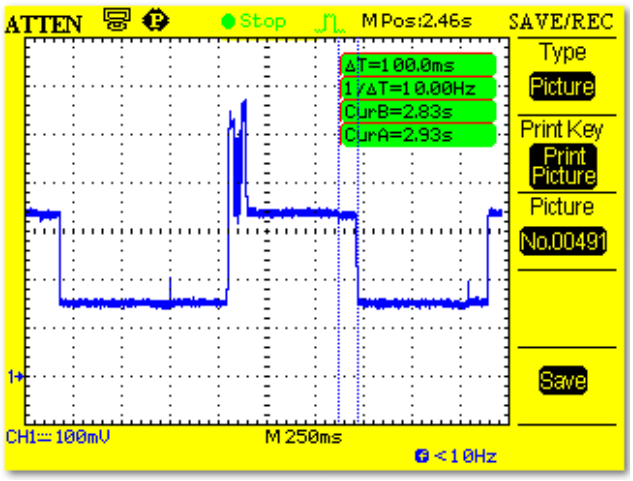


Fig. C.15.c. Proceso a Off

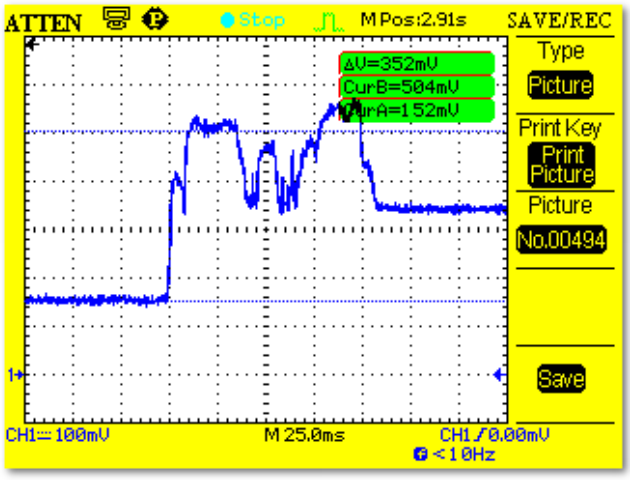


Fig. C.15.d. Pico encendido de la tarjeta

Cabe comentar que, aunque sepamos, aplicando la función `millis()`, el tiempo de ejecución de cada acción, no sabemos en qué punto empieza y acaba cada una de ellas, por lo tanto, para distinguirlo, se añaden una serie de funciones antes y después de la acción que se ejecuta (se verán en el ejemplo comentadas, ya que solo se utiliza para tal fin, no para ser medidas). Estas funciones lo que hacen es poner el pin digital 8 en alto y bajo para, así forzar un consumo extra que pueda ser observado con facilidad en el osciloscopio y facilitar mejor la distinción entre dónde y dónde va a durar cada acción:

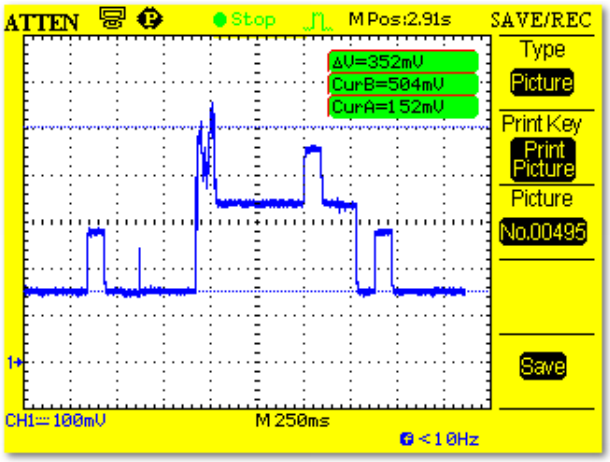


Fig. C.15.e. Poniendo el pin digital en alto y bajo

En la siguiente tabla se muestra el consumo de los diferentes procesos, de acuerdo al código que se ha ejecutado:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	520	7.78	4044
Estado On	500	18.4	9200
Proceso Off	100	18	1800

Tabla C.17. Consumo del proceso y estado on y proceso off de la tarjeta micro-SD

Se optó por poner un delay de 500 ms entre estado y estado para poder distinguir bien los procesosy que era suficiente para tal fin. Pero siguiendo el mismo proceso para todos, se toma como referencia tanto para el estado On como el Off de todos los estados "indefinidos" una duración de 1000 ms para que todos duren igual y sean exactos. Con lo cual, el consumo en el estado On sería:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Estado On	1000	18.4	18400

Tabla C.18. Consumo del estado on con 1000 ms de la tarjeta micro-SD

C.1.6.2. Escritura en la tarjeta

En este ejemplo se va a medir como escribir datos de 100 bytes en un fichero SD indicando la posición para escribir.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\SD_write_3

Se muestra las gráficas como resultado del ejemplo ejecutado:

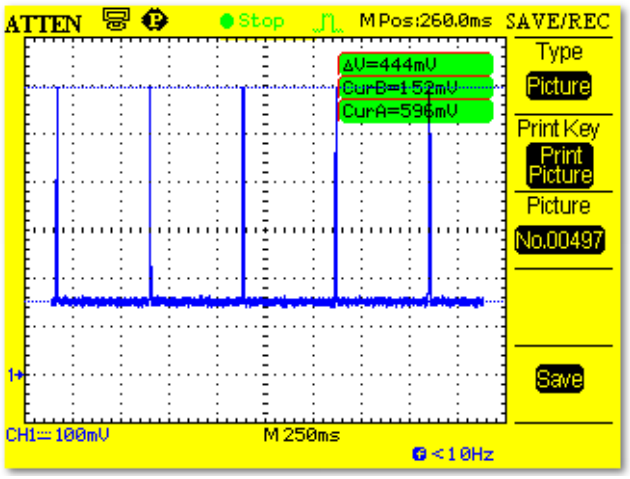


Fig. C.16.a. Medida general

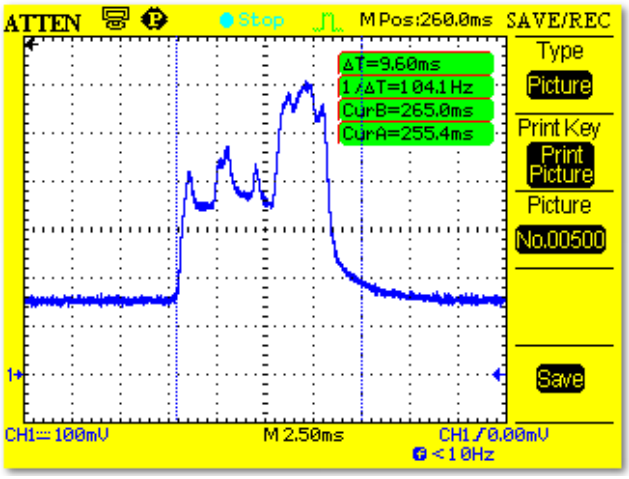


Fig. C.16.b. Tiempo de ejecución de la escritura

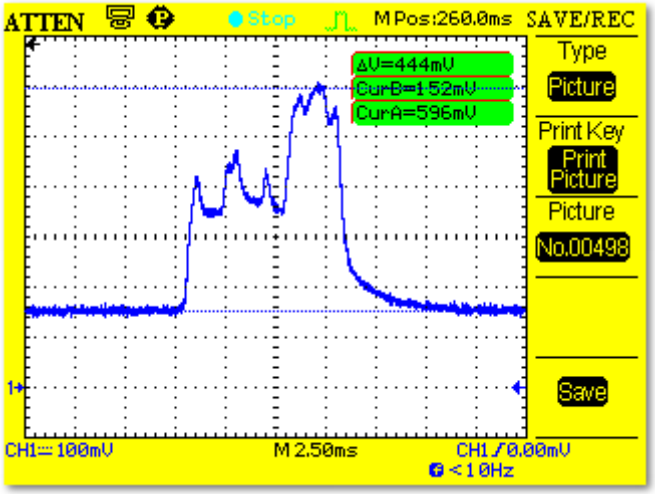


Fig. C.16.c. Consumo pico máximo

Como se ve en la gráfica (a), la lectura de la tarjeta de memoria SD no la hace desde el estado on de la misma.

En la siguiente tabla se muestra el consumo del proceso. Para ello, se ha dejado encendida la SD en el *setup* y así diferenciar cada proceso y observar con facilidad cual es el consumo de la escritura en la SD:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Escritura SD	9.6	27.77	266.68

Tabla C.19. Consumo de la escritura de la tarjeta micro-SD

C.1.6.3. Lectura de la tarjeta:

En este ejemplo se muestra como añadir datos al final del archivo SD y después mostrar como leer 100 bytes.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\funcionalidades del Waspmote\sd_read

Se muestra las gráficas como resultado del ejemplo ejecutado:

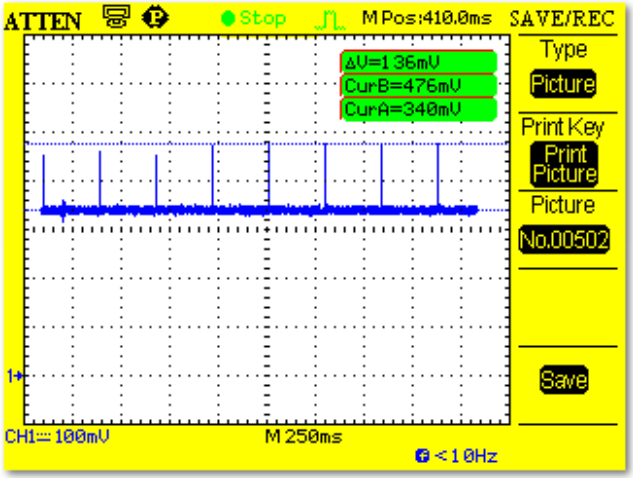


Fig. C.17.a. Medida general cada 300 ms

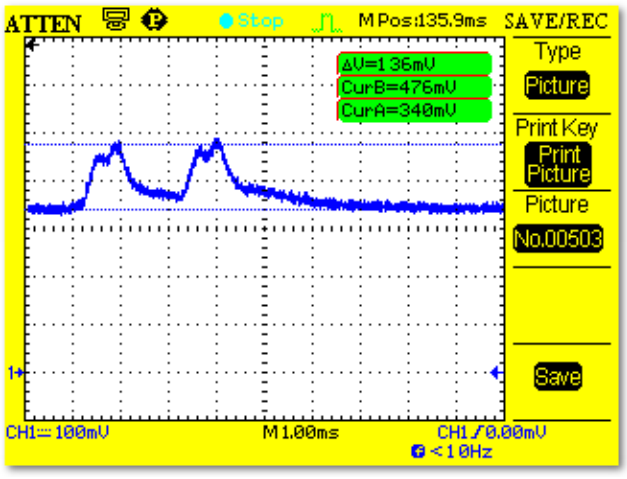


Fig. C.17.a. Proceso de lectura en la SD

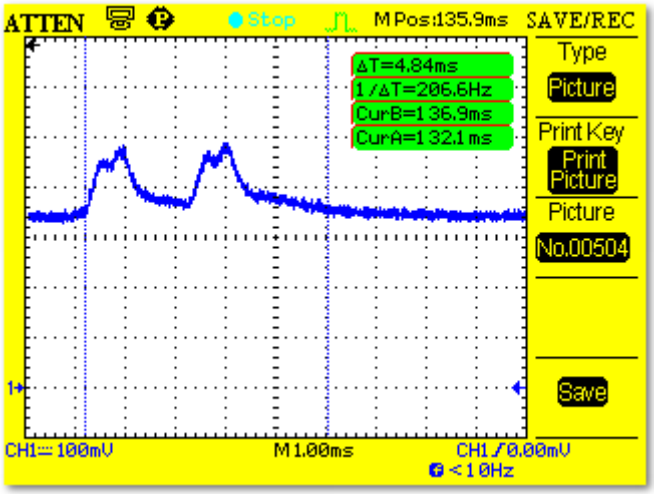


Fig. C.17.c. Tiempo de ejecución de la lectura

En la siguiente tabla se muestra el consumo del proceso. Para ello, se ha dejado encendida la SD en el *setup* y así diferenciar cada proceso y observar con facilidad cual es el consumo de la lectura en la SD:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Lectura SD	4.84	6.47	31.32

Tabla C.20. Consumo de la lectura de la tarjeta micro-SD

C.2. Placas integradas en la plataforma junto con la gama de sensores que se incluyen en cada una de ellas.

C.2.1. Placa de Agricultura.

C.2.1.1. Proceso On y Off de la placa

En este ejemplo se va a medir el proceso On y Off de la placa de agricultura.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\Agricultura_on_off

Se muestra las gráficas como resultado del ejemplo ejecutado.

(poner explicación de el por qué se forman los picos)

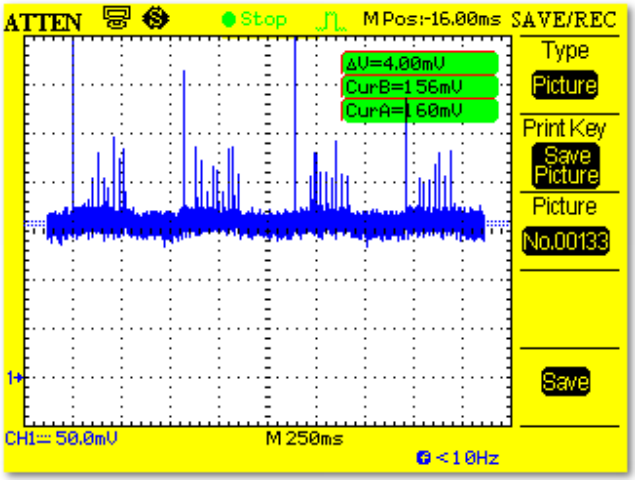


Fig. C.18.a. Medida general cada 300 ms

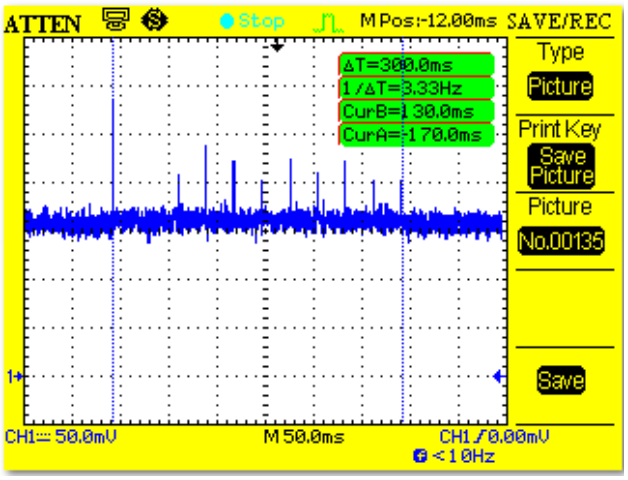


Fig. C.18.b. Estado On de la placa

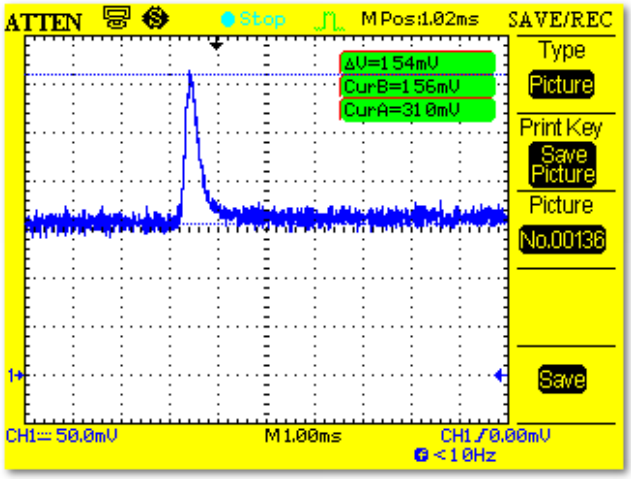


Fig. C.18.c. Corriente del pico del On

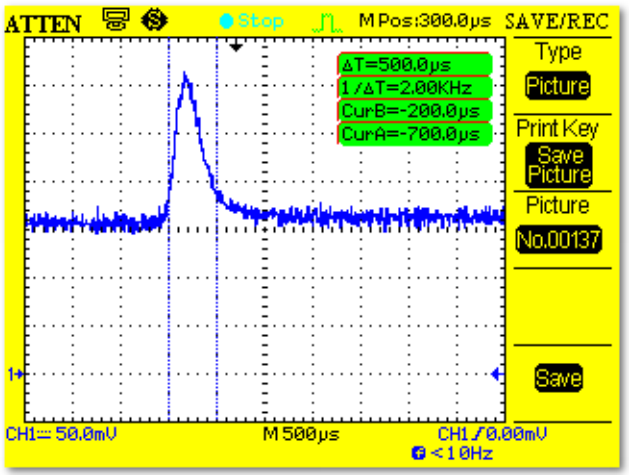


Fig. C.18.d. Tiempo de ejecución del On

En la siguiente tabla se muestra el consumo del proceso y estado On y proceso Off:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	0.5	10.78	5.39
Estado On	300	0.455	136.61
Proceso Off	0	0.455	0

Tabla C.21. Consumo del proceso y estado on y proceso off de la plada de Agricultura

Se optó por poner un delay de 300 ms entre estado y estado. Pero siguiendo el mismo proceso para todos, se toma como referencia tanto para el estado On como el Off de todos los estados “indefinidos” una duración de 1000 ms para que todos duren igual y sean exactos. Con lo cual, el consumo en el estado On sería:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Estado On	1000	0.46	460

Tabla C.22. Consumo del estado on con 1000 ms de la placa de Agricultura

C.2.1.2. Lectura del sensor de presión atmosférica

En este ejemplo se va a medir la lectura del sensor de presión atmosférica. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_presion_2

Se muestra las gráficas como resultado del ejemplo ejecutado:

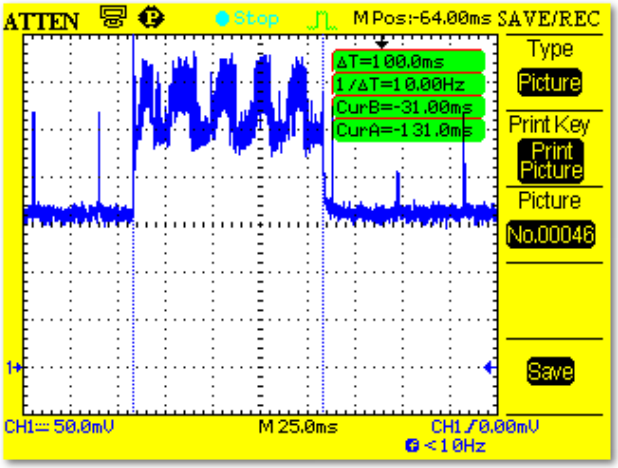


Fig. C.19.a. Tiempo de ejecución del proceso completo

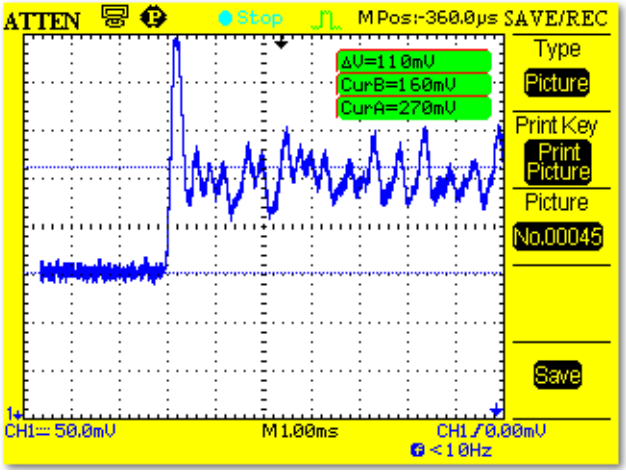


Fig. C.19.b. Captura del Inicio del proceso

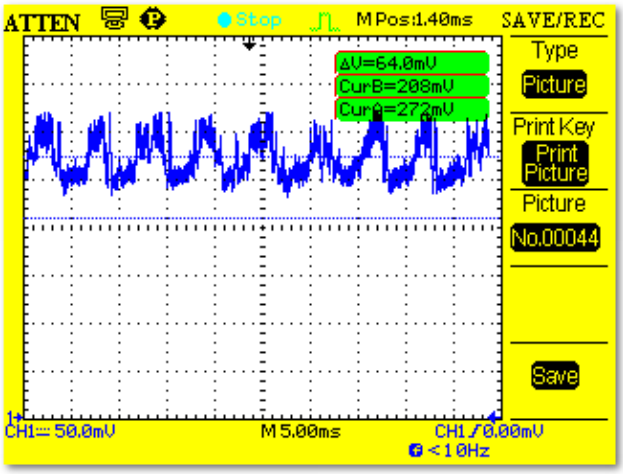


Fig. C.19.c. Captura durante el proceso

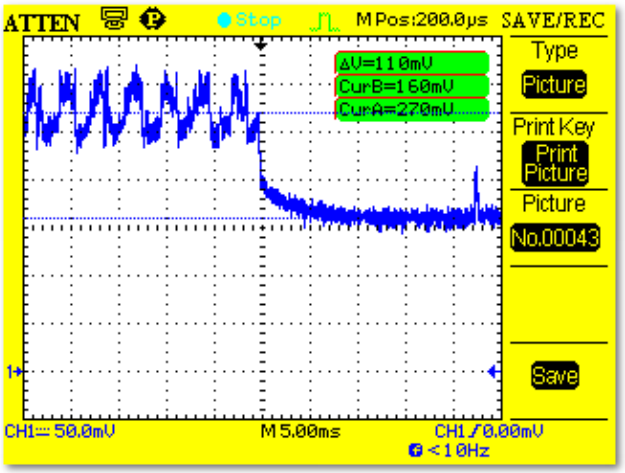


Fig. C.19.d. Captura del final del proceso

En la siguiente tabla se muestra el consumo de todo el proceso, ya que para los sensores no se calcula el consumo de cada función ejecutada por separado como se ha hecho con los ejemplos anteriores, si no que se calcula el proceso completo de encendido, lectura y apagado del sensor (se aplica para el resto de sensores). Para ello, se ha dejado encendida la placa de agricultura en el *setup* y así diferenciar cada proceso y observar con facilidad cual es el consumo de la lectura del sensor:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	100	11	1100

Tabla C.23. Consumo de la lectura del sensor de presión atmosférica

Como se ve, la duración total del proceso corresponden con los 100 ms que se han puesto en el código después de encender el sensor y que son necesarios para que el sensor ofrezca una señal valida y precisa desde que se alimenta, debido a que a la señal de alimentación le lleva este tiempo estabilizarse.

C.2.1.3. Lectura del sensor de humedad

En este ejemplo se va a medir la lectura del sensor de humedad. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_humedad

Se muestra las gráficas como resultado del ejemplo ejecutado:

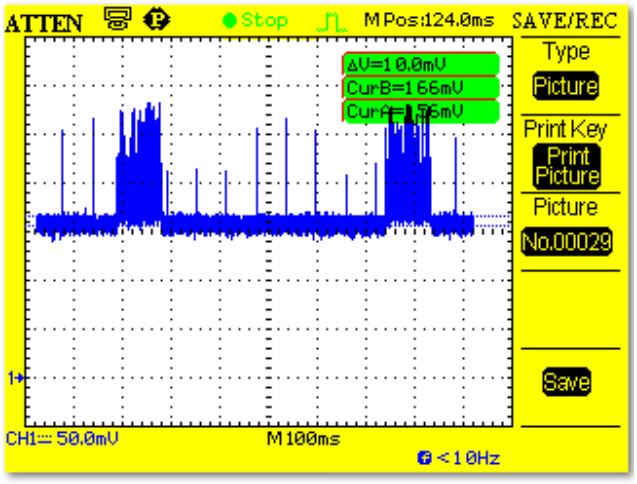


Fig. C.20.a. Medida del proceso completo cada 500 ms

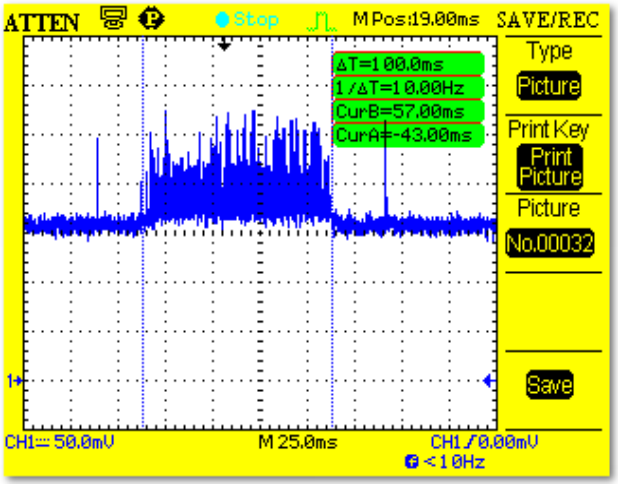


Fig. C.20.b. Tiempo de ejecución del proceso completo

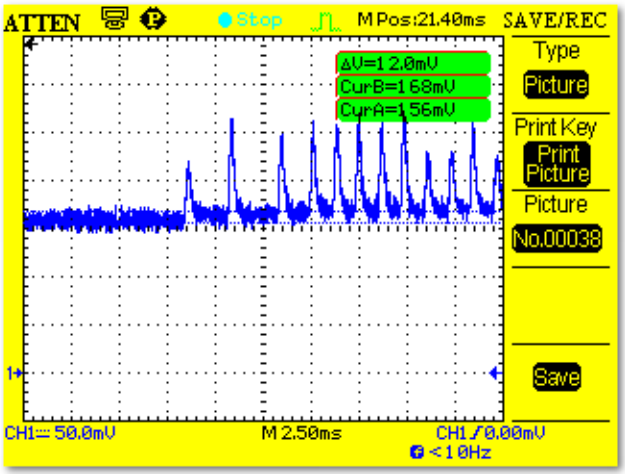


Fig. C.21.c. Inicio del proceso

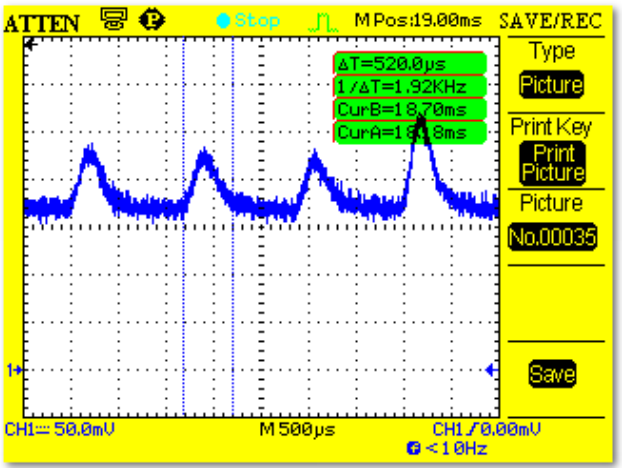


Fig. C.21.d. Duración de cada pico de lectura

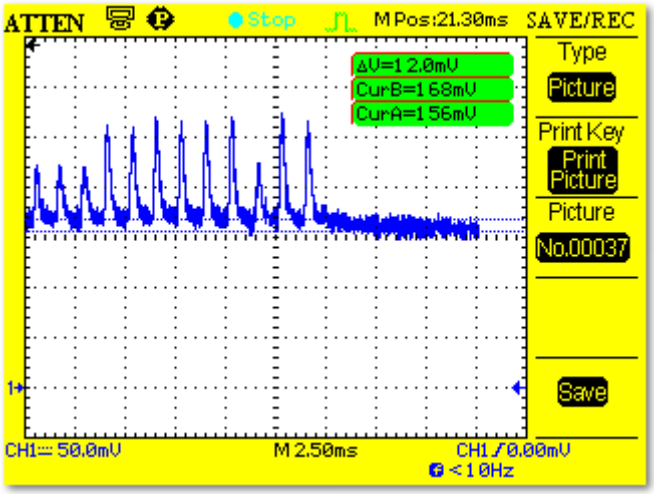


Fig. C.21.e. Final del proceso

En la siguiente tabla se muestra el consumo de todo el proceso:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	100	3.22	322.25

Tabla C.24. Consumo de la lectura del sensor de humedad

C.2.1.4. Lectura del sensor LDR

En este ejemplo se va a medir la lectura del sensor LDR. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_ldr

Se muestra las gráficas como resultado del ejemplo ejecutado:

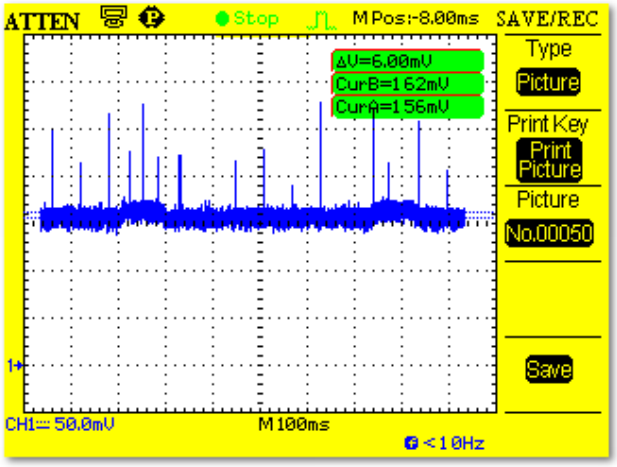


Fig. C.22.a. Medida general cada 500 ms

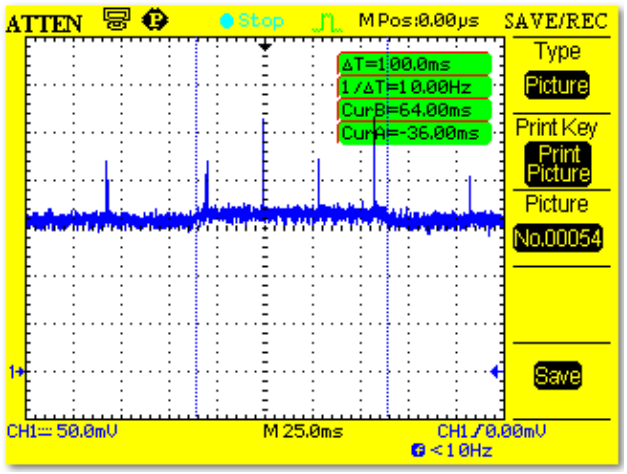


Fig. C.22.b. Tiempo de ejecución del proceso

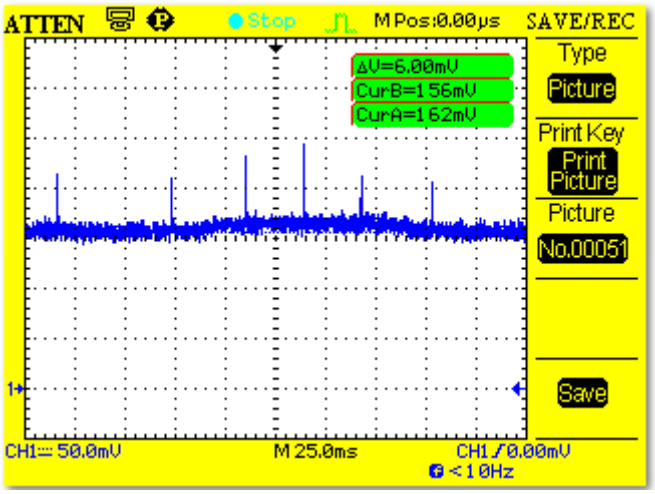


Fig. C.22.c. Proceso completo

En la siguiente tabla se muestra el consumo de todo el proceso:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	100	0.340	34

Tabla C.25. Consumo de la lectura del sensor LDR

C.2.1.5. Lectura del Sensirion

En este ejemplo se va a medir la lectura del sensirion. Para ello, hace falta previamente

encender dicho sensor. Como se incluyen dos tipos de sensores, temperatura y humedad, primero se leerá el de temperatura seguida por el de humedad.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensirion

Se muestra las gráficas como resultado del ejemplo ejecutado:

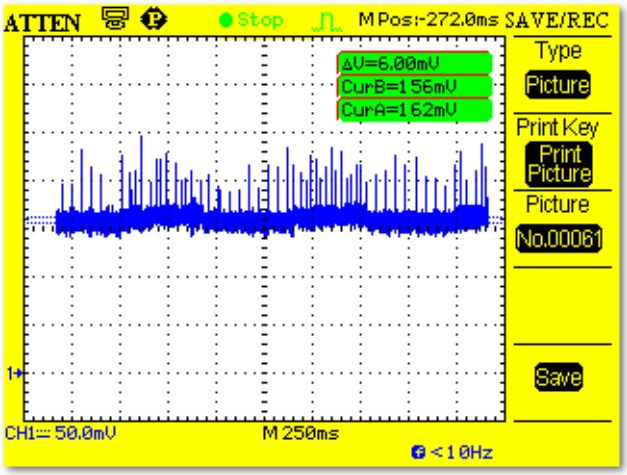


Fig. C.23.a. Medida general cada 500 ms

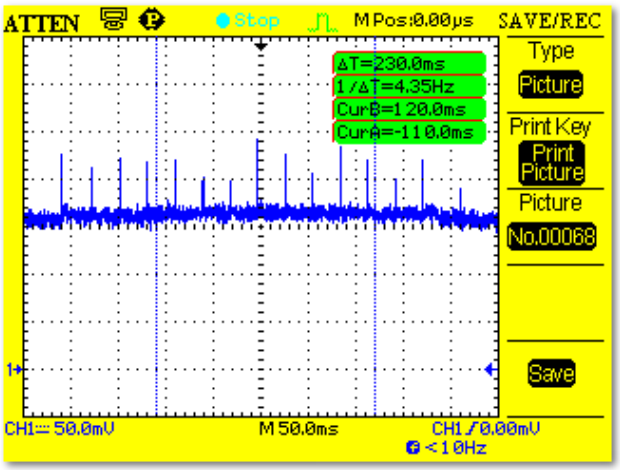


Fig. C.23.b. Tiempo de ejecución de la lectura del sensor de Tª

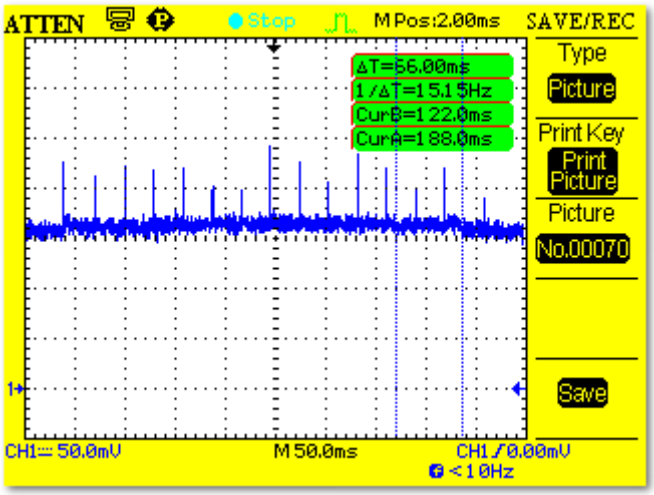


Fig. C.23.c. Tiempo de ejecución de la lectura del sensor de humedad

Se observa en las gráficas, que el tiempo que tardar en ejecutarse la función de lectura del sensor de temperatura y el de humedad es 230 ms y 66 ms, respectivamente.

En la siguiente tabla se muestra el consumo del proceso completo:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	396	0.424	168

Tabla C.26. Consumo de la lectura del sensor LDR

C.2.1.6. Lectura del sensor Watermark

En este ejemplo se va a medir la lectura del Watermark. Para ello, hace falta encender dicho sensor. Cabe destacar, que las pruebas se realizó mojándolo previamente ya que sin mojar, la lectura del sensor costaba mucho más que este, es decir unos 41500 ms, mientras que si lo mojas son 240 ms y había que seguir una medida más realista, con lo cual se optó por lo segundo.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_watermark

Se muestra las gráficas como resultado del ejemplo ejecutado:

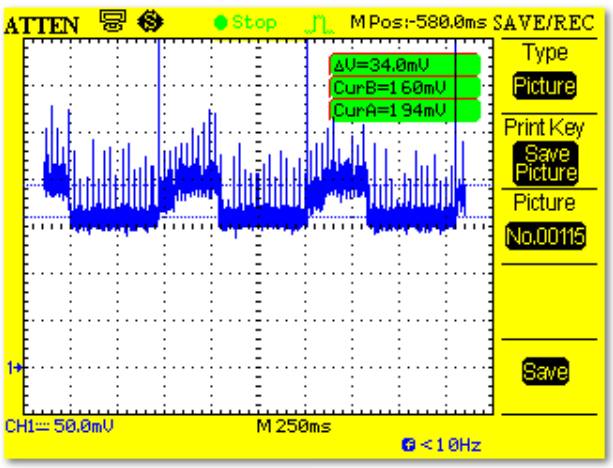


Fig. C.24.a. Medida general cada 500 ms

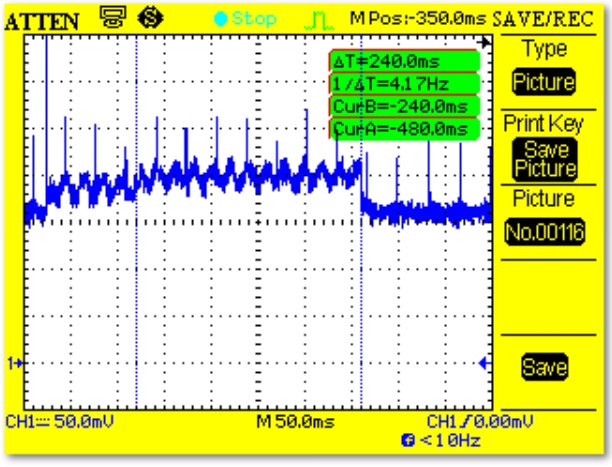


Fig. C.24.b. Tiempo de ejecución de la lectura del sensor

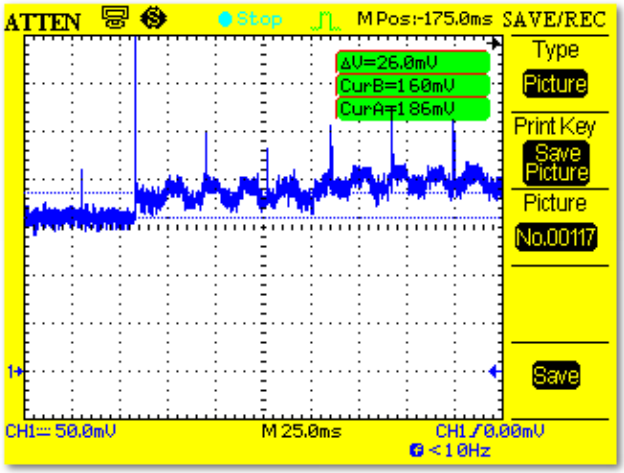


Fig. C.24.c. Inicio del proceso

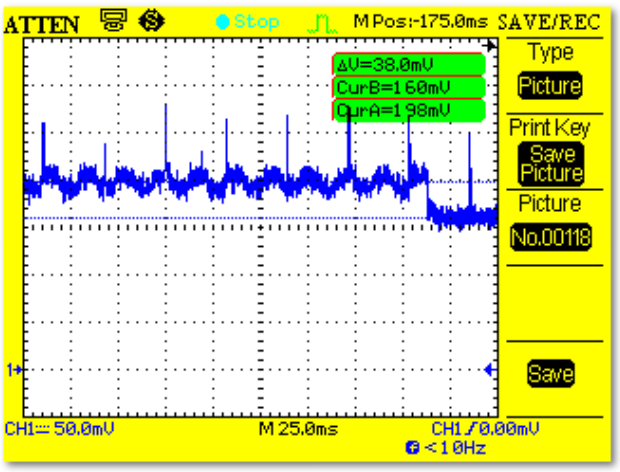


Fig. C.24.d. Consumo durante el proceso

En la siguiente tabla se muestra el consumo del proceso completo:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	340	3.44	1172

Tabla C.27. Consumo de la lectura del sensor watermark mojado

Comentar que, se hizo unas pruebas con el Watermark seco, cuyo tiempo de ejecución de la lectura del sensor era de 39784 ms a 2.8 mA de consumo de corriente. Con lo cual es consumo total del proceso era:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	39884	2.8	111675.2

Tabla C.28. Consumo de la lectura del sensor watermark seco

C.2.1.7. Estación meteorológica

Lectura Anemómetro:

En este ejemplo se muestra como leer el anemómetro, sensor que forma parte de la estación meteorológica. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\estaci_n_meteorol_gica_anem_metro

Se muestra las gráficas como resultado del ejemplo ejecutado:

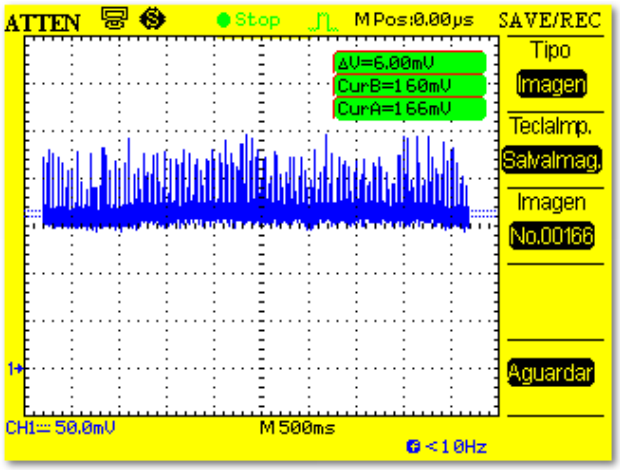


Fig. C.25.a. Medida general cada 500 ms

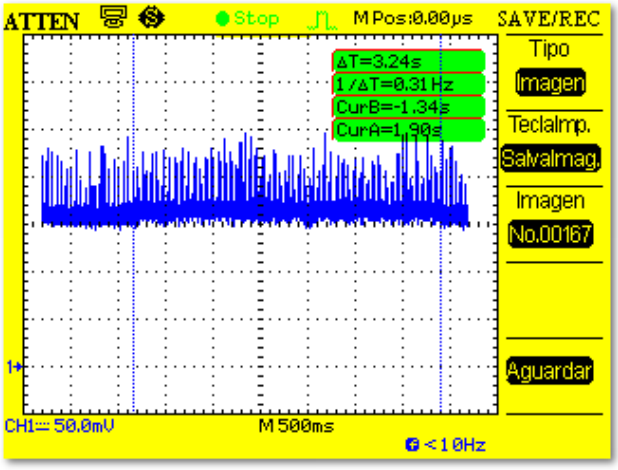


Fig. C.25.b. Tiempo de ejecución del proceso

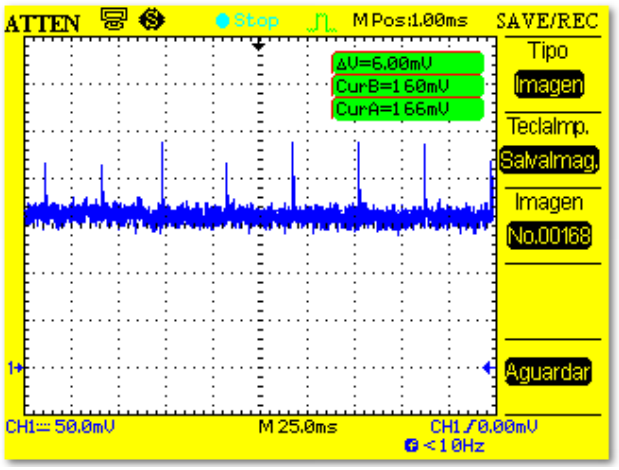


Fig. C.25.c. Final del proceso

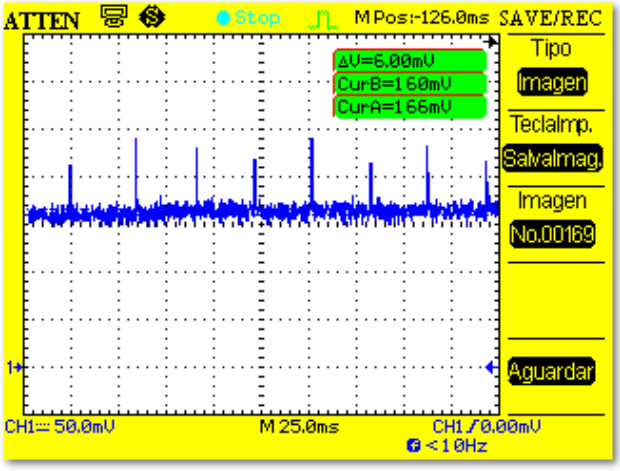


Fig. C.25.d. Inicio del proceso

En la gráfica b, se ve con diferencia que el tiempo de lectura del anemómetro es de 3140 ms.

En la siguiente tabla se muestra el consumo del proceso completo:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	3240	0.6	1944

Tabla C.29. Consumo de la lectura del anemómetro

Lectura Veleta:

En este ejemplo se muestra como leer la veleta, sensor que forma parte de la estación meteorológica. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\estacion_meteorologica_veleta

Se muestra las gráficas como resultado del ejemplo ejecutado:

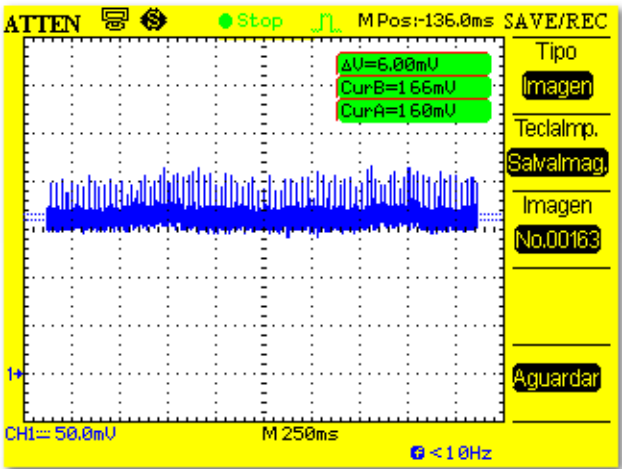


Fig. C.26.a. Medida general cada segundo

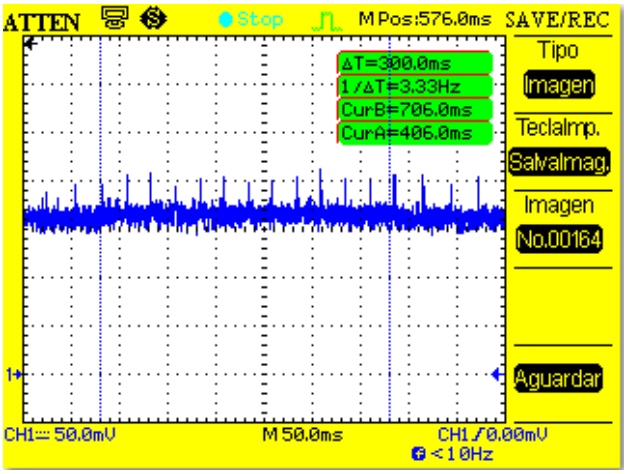


Fig. C.26.b. Tiempo de ejecución del proceso

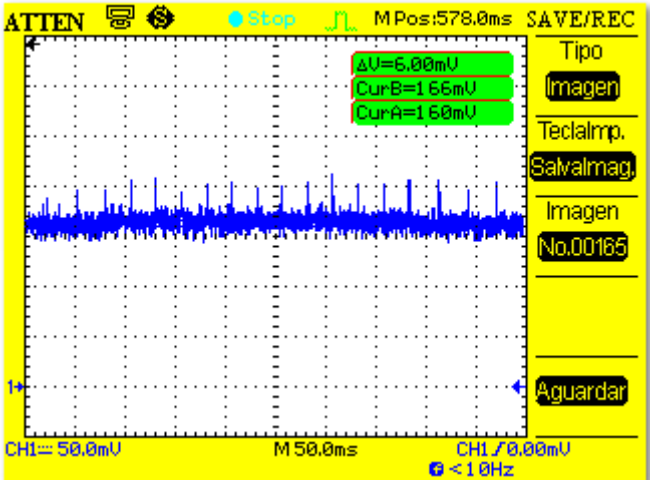


Fig. C.26.c. Consumo del proceso

En la gráfica b, se ve con diferencia que el tiempo de lectura del anemómetro es de 205 ms.

En la siguiente tabla se muestra el consumo del proceso completo:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	305	0.6	183

Tabla C.30. Consumo de la lectura de la veleta

Interrupción del pluviómetro:

Este ejemplo muestra como habilitar la interrupción del pluviómetro de la estación meteorológica de la placa de Agricultura.

Detecta el número de pulsos del pluviómetro y luego regresa a sleep.

A continuación, se muestra las gráficas como resultado del ejemplo ejecutado:

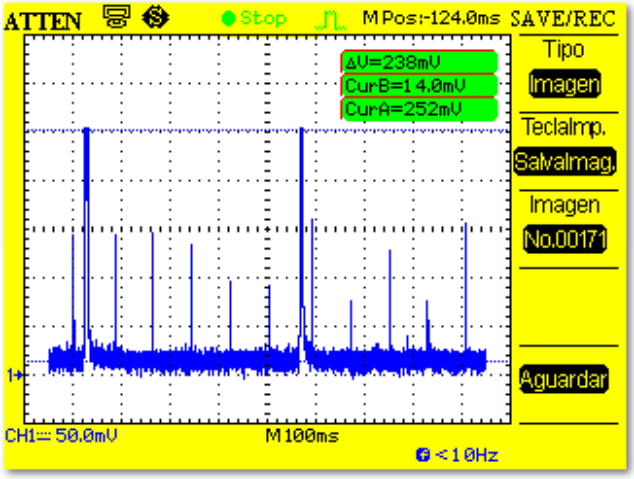


Fig. C.27.a. Medida general

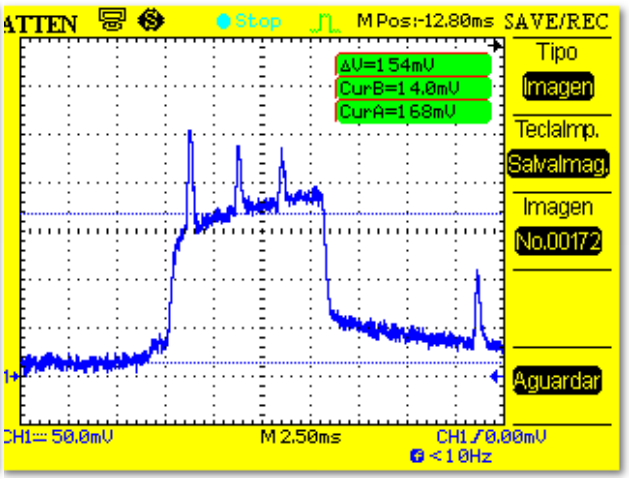


Fig. C.27.b. Proceso durante la interrupción

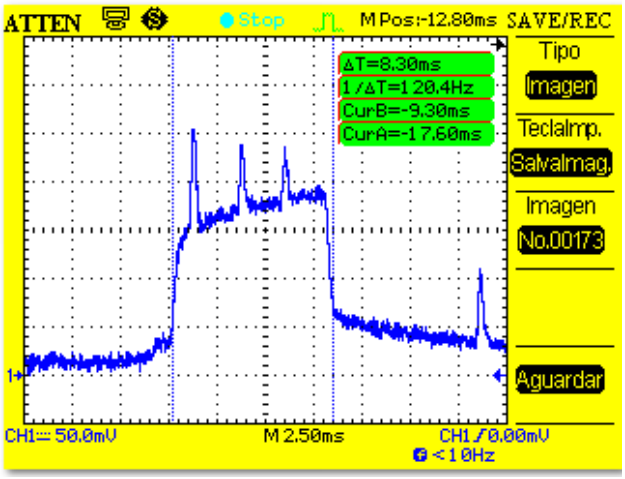


Fig. C.27.c. Tiempo de ejecución del proceso durante la interrupción

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Interrupción pluviómetro	8.30	1.37	11.38

Tabla C.31. Consumo de la interrupción del pluviómetro

C.2.2. Placa de Agricultura PRO

La placa de agricultura PRO se ha diseñado de tal forma para que cuando enciendes y se apaguen los sensores, se enciende y se apague la placa automáticamente respectivamente. Con lo cual, el consumo que va a aparecer va a ser el total, es decir la placa de agricultura PRO más el sensor correspondiente.

C.2.2.1. Proceso On y off de la placa

En este ejemplo se va a medir el proceso On y Off de la placa de agricultura versión PRO.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\Agricultura_on_off

Se muestra las gráficas como resultado del ejemplo ejecutado:

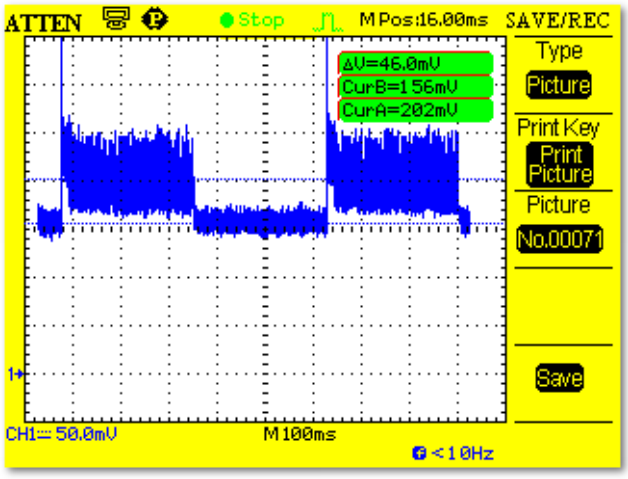


Fig. C.28.a. Medida general cada 300 ms

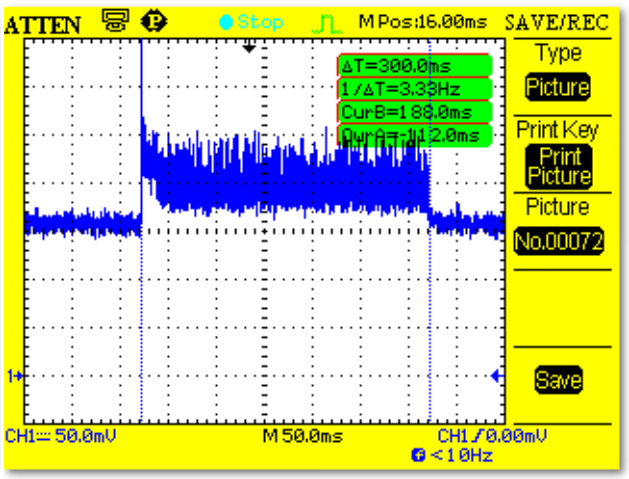


Fig. C.28.b. Estado On de la placa

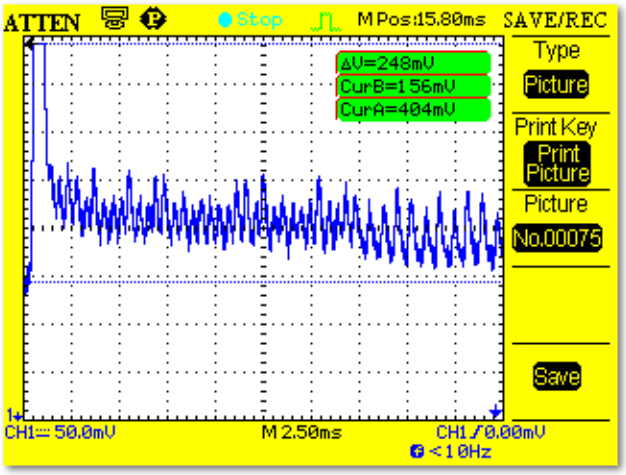
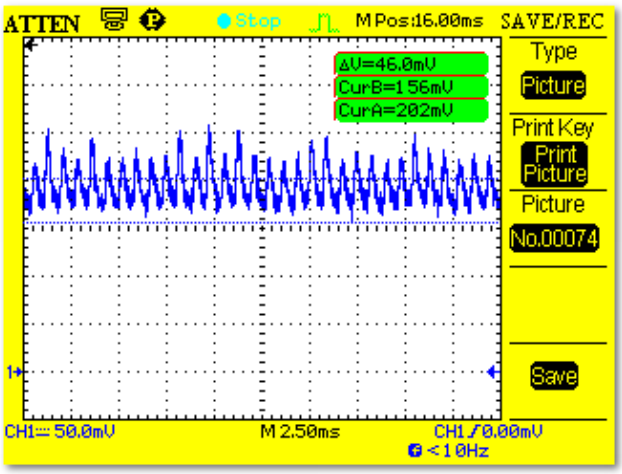


Fig. C.28.c. Inicio del estado On Fig.



C.28.d. Consumo durante el estado On

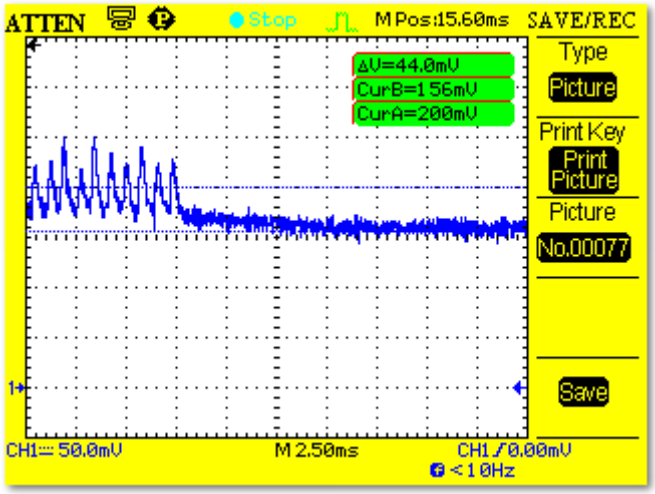


Fig. C.28.e. Final del estado On

En la siguiente tabla se muestra el consumo del proceso y estado On y proceso Off:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	1	24.8	24.8
Estado On	300	4.6	1380
Proceso Off	0	4.6	0

Tabla C.32. Consumo del proceso y estado on y proceso off de la placa de Agricultura PRO

Se optó por poner un delay de 300 ms entre estado y estado. Pero siguiendo el mismo proceso para todos, se toma como referencia tanto para el estado On como el Off de todos los estados “indefinidos” una duración de 1000 ms para que todos duren igual y sean exactos. Con lo cual, el consumo en el estado On sería:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Estado On	1000	4.6	4600

Tabla C.33. Consumo del estado on con 1000 ms de la placa de Agricultura PRO

Se ve claramente que hay una gran diferencia de consumo en esta versión de la placa de Agricultura respecto a la normal.

C.2.2.2. Lectura del sensor PT-1000

En este ejemplo se va a medir la lectura del sensor PT-1000. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_pt1000

Se muestra las gráficas como resultado del ejemplo ejecutado:

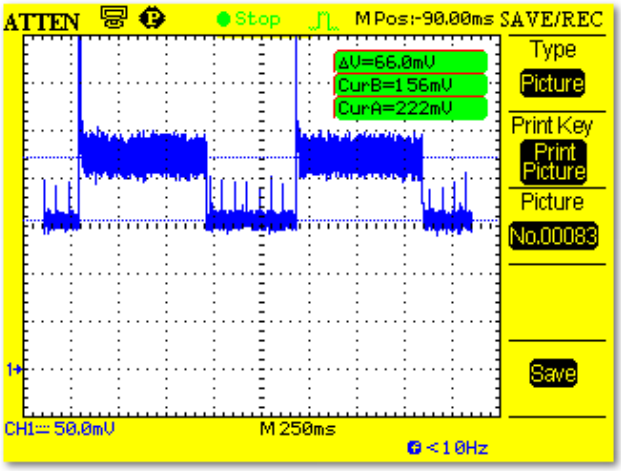


Fig. C.29.a. Medida general cada 500 ms

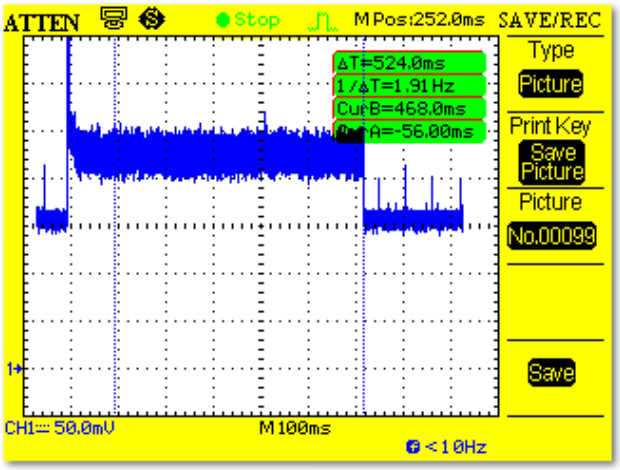


Fig. C.29.b. Tiempo de ejecución de la lectura del sensor

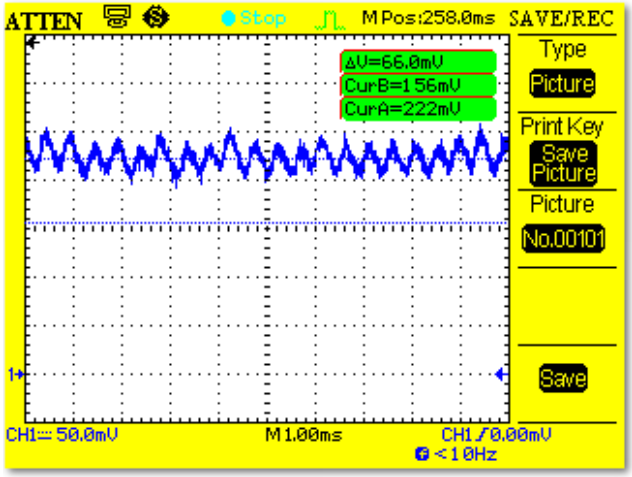


Fig. C.29.c. Consumo durante el proceso de lectura

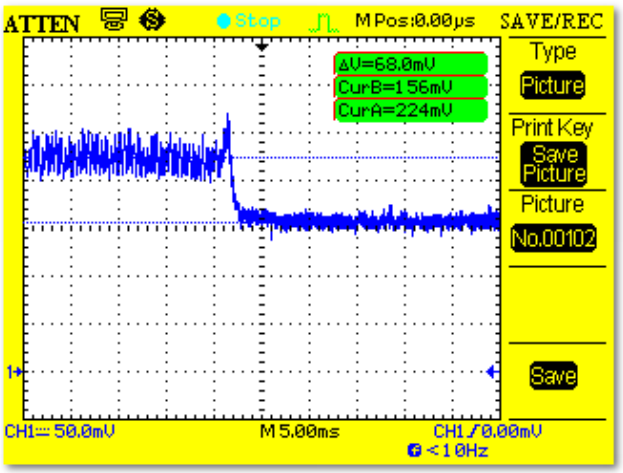


Fig. C.29.d. Final del proceso

Como se observa en la gráfica (b), 524 ms es el tiempo que le cuesta leer el sensor. Como se observa en las gráficas.

Sabiendo el consumo de la placa, calculado en el punto anterior, es fácil sacar lo que consume leer el sensor:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	624	2	1248

Tabla C.34. Consumo de la lectura del sensor PT1000

C.2.2.3. Lectura del dendrómetro DF

En este ejemplo se va a medir la lectura del dendrómetro DF. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_dendrometro

Se muestra las gráficas como resultado del ejemplo ejecutado:

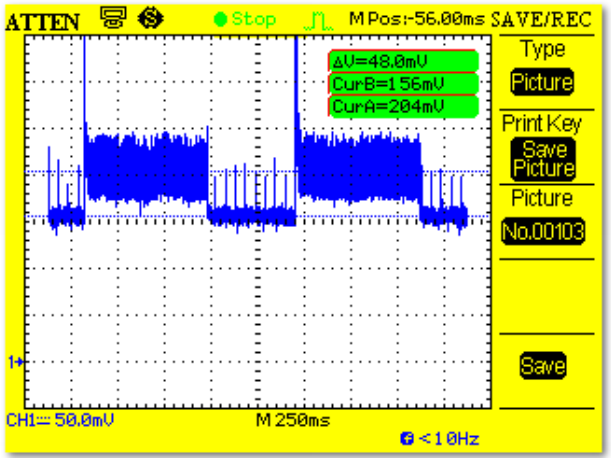


Fig. C.30.a. Medida general cada 500 ms

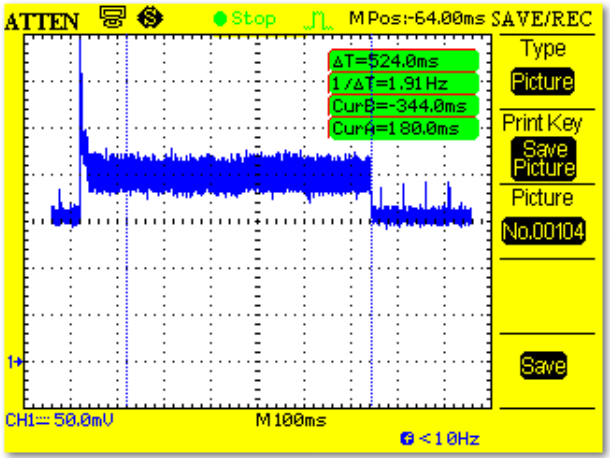


Fig. C.30.b. Tiempo de ejecución de la lectura del sensor

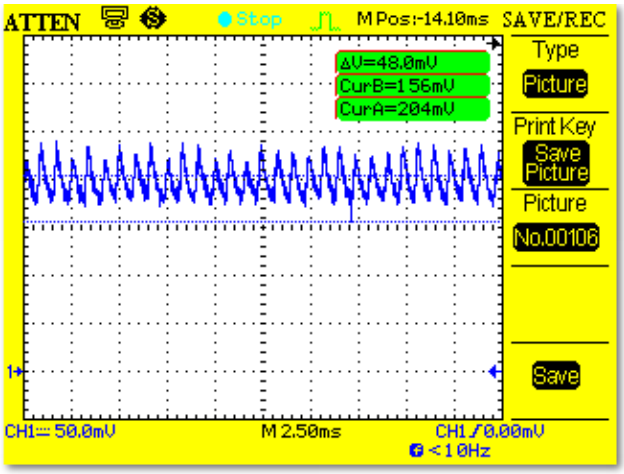


Fig. C.30.c. Consumo durante el proceso de lectura

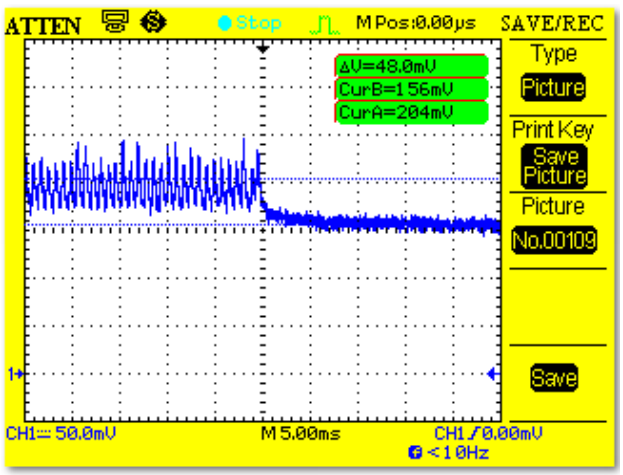


Fig. C.30.d. Final del proceso

Como se observa en la gráfica (b), 524 ms es el tiempo que le cuesta leer el sensor.

Sabiendo el consumo de la placa, es fácil sacar lo que consume leer el sensor:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	624	0.2	124.8

Tabla C.35. Consumo de la lectura del dendrómetro DF

C.2.2.4. Lectura del dendrómetro DC2

En este ejemplo se va a medir la lectura del dendrómetro DC2. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra el *link* que lleva al código ejecutado:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_dendrometro

Se muestra las gráficas como resultado del ejemplo ejecutado:

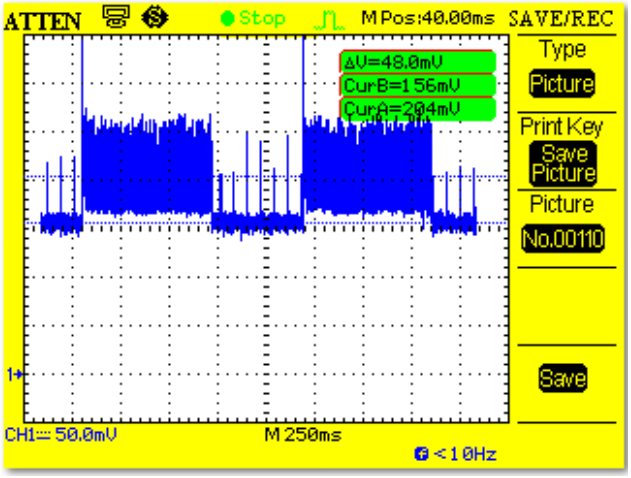


Fig. C.31.a. Medida general cada 500 ms

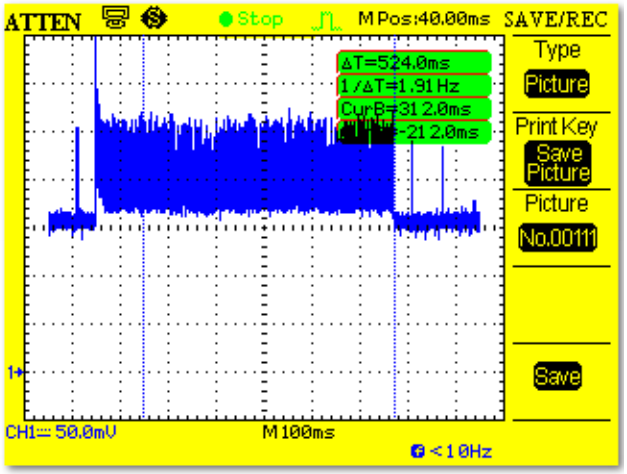


Fig. C.31.b. Tiempo de ejecución de la lectura del sensor

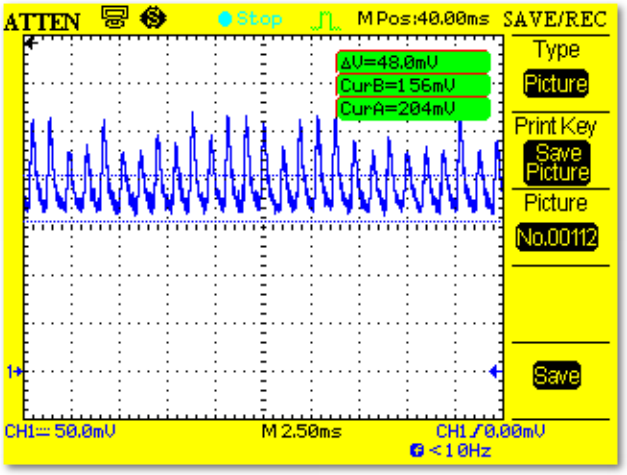


Fig. C.31.c. Consumo durante el proceso de lectura

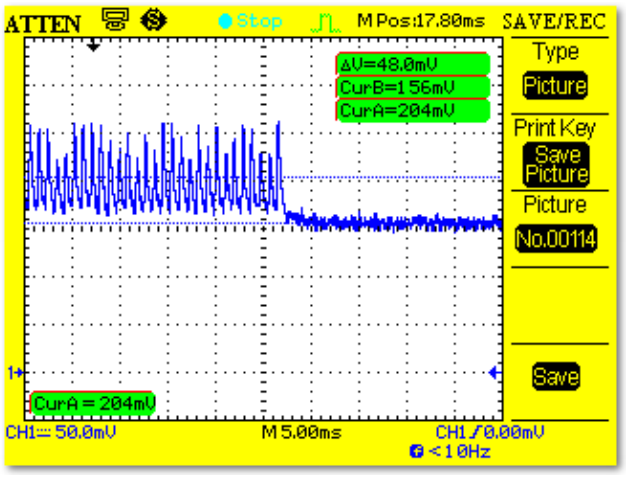


Fig. C.31.d. Final del proceso

Como se observa en la gráfica (b), 524 ms es el tiempo que le cuesta leer el sensor.

Sabiendo el consumo de la placa, es fácil sacar lo que consume leer el sensor:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	624	0.2	124.8

Tabla C.36. Consumo de la lectura del dendrómetro DC2

C.2.2.5. Lectura del sensor SQ-110 y SU-100

En este ejemplo se va a medir la lectura de los sensores SQ-110 y SU-100. Para ello, hace falta previamente encender dicho sensor.

A continuación, se muestra los *links* que llevan a los códigos ejecutados. El primero corresponde al SQ-110 y el segundo al SU-100:

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_radiacion_solar

3_sw\3_01_codigos_definitivos\3_01_01_ejemplos\sensores\Agricultura\sensor_radiacion_ultravioleta

Se muestra las gráficas como resultado de los ejemplos ejecutados. En este caso, al consumir lo mismo, son las mismas para los dos:

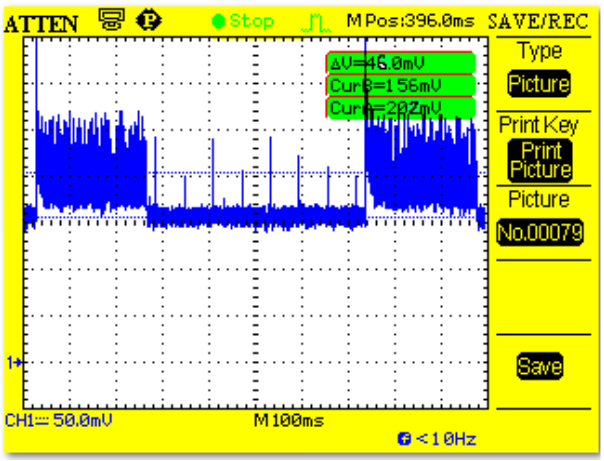


Fig. C.32.a. Medida general cada 500 ms

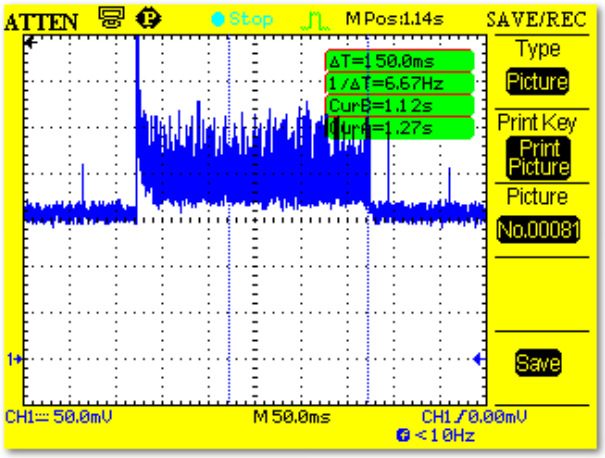


Fig. C.32.b. Tiempo de ejecución de la lectura del sensor

Como se observa en la gráfica (b), 150 ms es el tiempo que le cuesta leer el sensor aunque el consumo del sensor es 0 mA.

Sabiendo el consumo de la placa, es fácil sacar lo que consume leer el sensor:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	250	0	0

Tabla C.37. Consumo de la lectura de los sensores SQ-110 y SU-100

C.2.3. Placa de Eventos

C.2.3.1. Proceso de encendido y apagado de la placa de Eventos

Este ejemplo muestra como encender y apagar la placa de Eventos con todos los interruptores activados.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

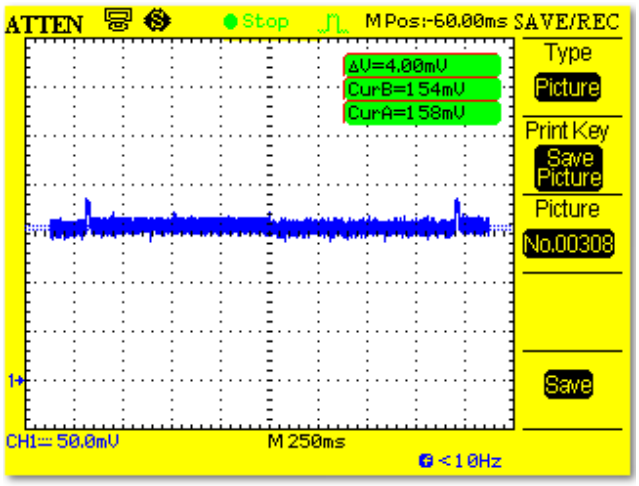


Fig. C.33.a. Medida general cada segundo

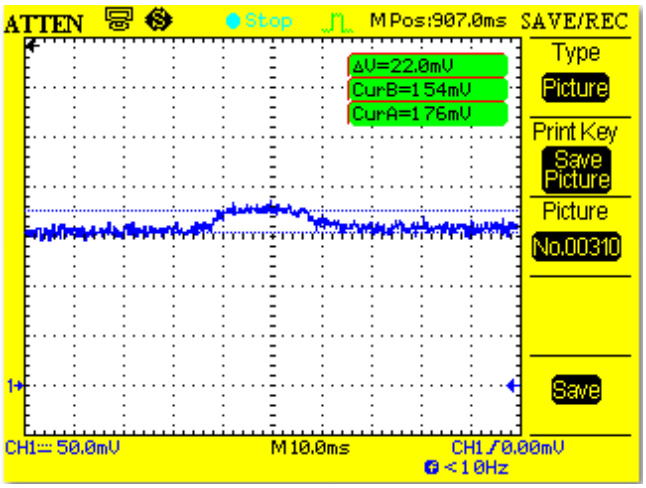


Fig. C.33.b. Pico de carga del proceso on

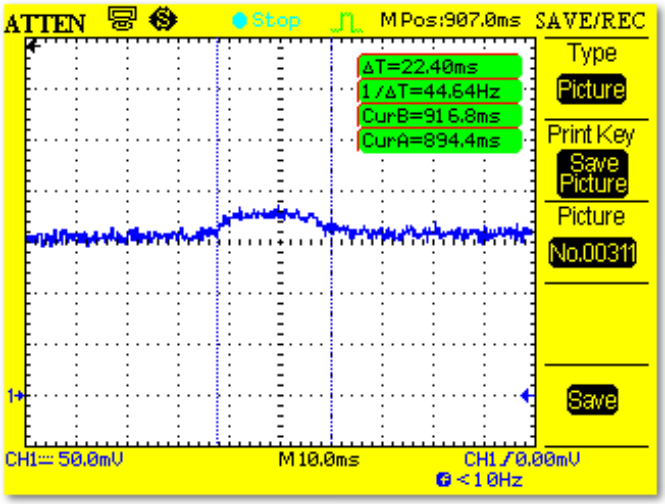


Fig. C.33.c. Tiempo de ejecución del proceso on

El consumo en el proceso y estado on y en el proceso off la placa de Eventos es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso On	22.4	1.76	39.424
Estado On	1000	0.3	300
Proceso Off	0	0.3	0

Tabla C.38. Consumo del proceso y estado on y del proceso off la placa de Eventos

C.2.3.2. Sensor de Presión (Flexiforce PS-02)

Este ejemplo muestra como leer el sensor Flexiforce PS-02 en el socket 1 de la placa de Eventos cuando se produce una interrupción. Para usar este ejemplo, el primer interruptor debe ser activado. Es necesario saber que el consumo en el estado on del primer interruptor de la placa de Eventos es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

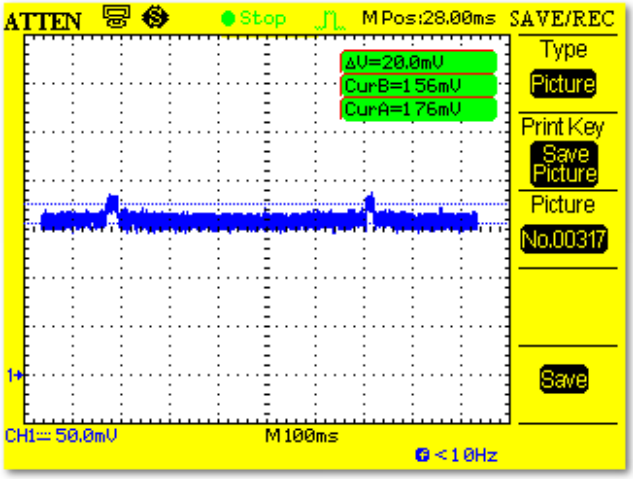


Fig. C.34.a. Medida general de lectura del sensor

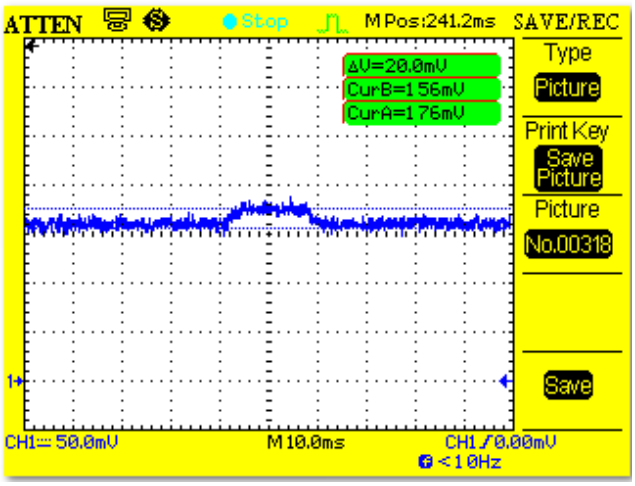


Fig. C.34.b. Consumo al producirse la interrupción

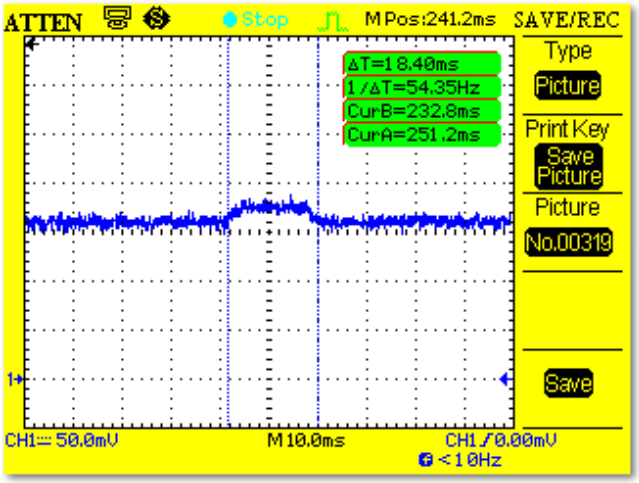


Fig. C.34.c. Tiempo de ejecución de la interrupción

Cuando se produce una interrupción en el sensor y se ejerce fuerza, se produce ese pico que se ve en las gráficas.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura Flexiforce PS-02	28.4	1.02	29.018

Tabla C.39. Consumo de la lectura del sensor Flexiforce PS-02

C.2.3.3.- Sensor de doblaje (FLX-C1-H)

Este ejemplo muestra como leer el sensor FLX-01-H en el socket 2 de la placa de Eventos cuando se produce una interrupción. Para usar este ejemplo, se debe activar el segundo interruptor. Es necesario saber que el consumo en el estado on de la placa de Eventos con el segundo interruptor activado es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

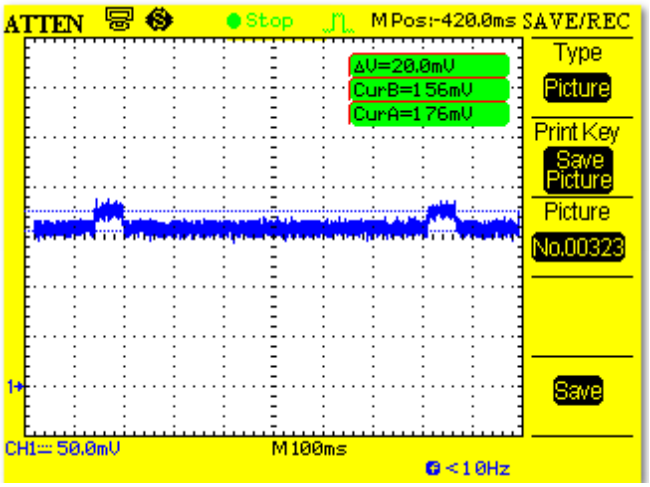


Fig. C.35.a. Medida general de lectura del sensor

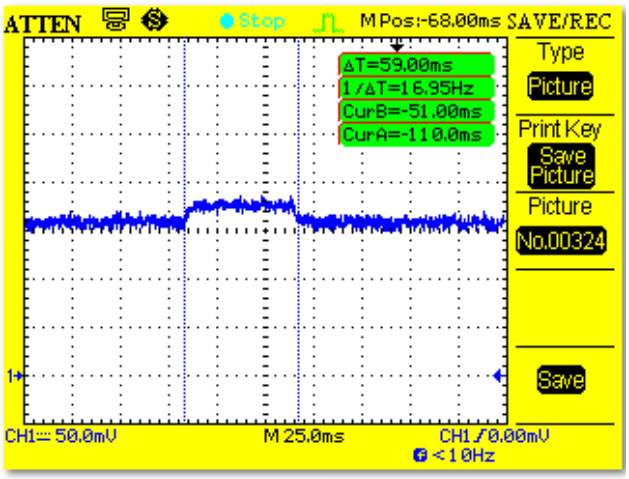


Fig. C.35.b. Tiempo de ejecución de la interrupción

Cuando se produce una interrupción en el sensor y se dobla, se produce ese pico que se ve en las gráficas.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura FLX-01-H	69	1.68	116.312

Tabla C.40. Consumo de la lectura del sensor FLX-01-H

C.2.3.4. Sensor de vibración PZ-01 y PZ-08

Este ejemplo muestra como leer los sensores de vibración en el socket 4 de la placa de Eventos cuando se produce una interrupción. Para usar este ejemplo, se debe activar el cuarto interruptor. Es necesario saber que el consumo en el estado on de la placa activando este interruptor, es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

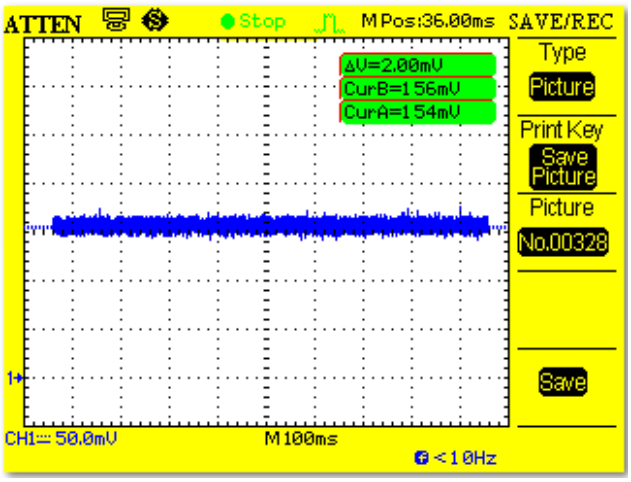


Fig. C.36. Medida general de lectura del sensor

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor vibración	10	0.2	2

Tabla C.41. Consumo de la lectura del sensor de vibración

C.2.3.5. Sensor de temperatura

Este ejemplo muestra como leer el sensor de temperatura en el socket 6 de la placa de Eventos. Para usar este ejemplo, se debe activar el sexto interruptor. Es necesario saber conocer que el consumo en el estado on de la placa de eventos activando este interruptor, es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

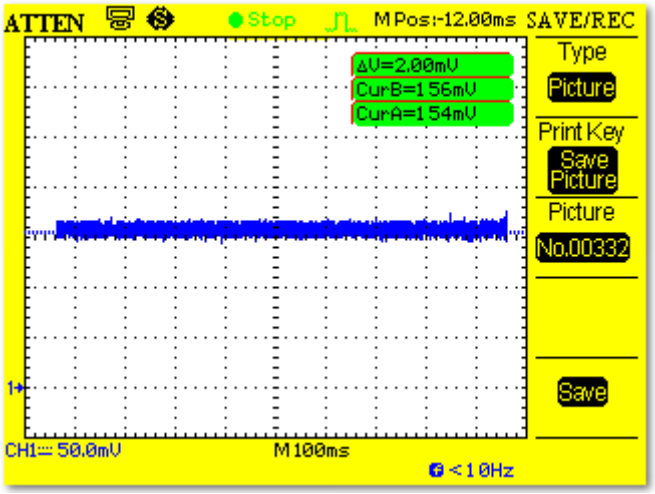


Fig. C.37. Medida general de lectura del sensor

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energetic total (mA·ms)
Lectura sensor temperatura	100	0.005	0.5

Tabla C.42. Consumo de la lectura del sensor de temperatura

C.2.3.6. Sensor de nivel de líquido PTFA3415

Este ejemplo muestra como leer el sensor PTFA3415 en el socket 1 de la placa de Eventos. Para usar este ejemplo, se debe activar el primer interruptor. Es necesario saber que el consume en el estado on de la placa activando este interruptor es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

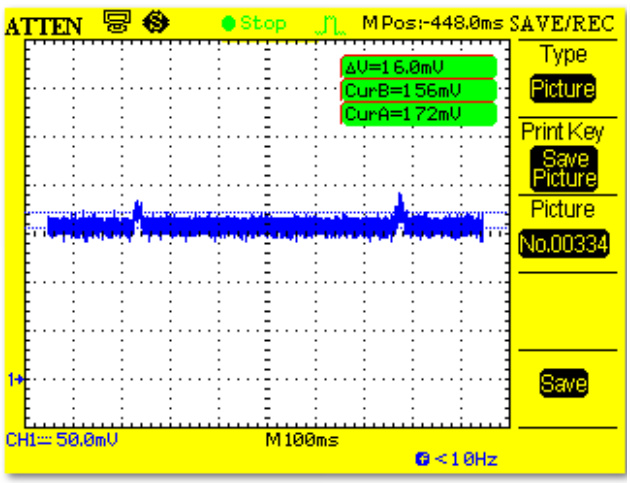


Fig. C.38.a. Medida general de lectura del sensor

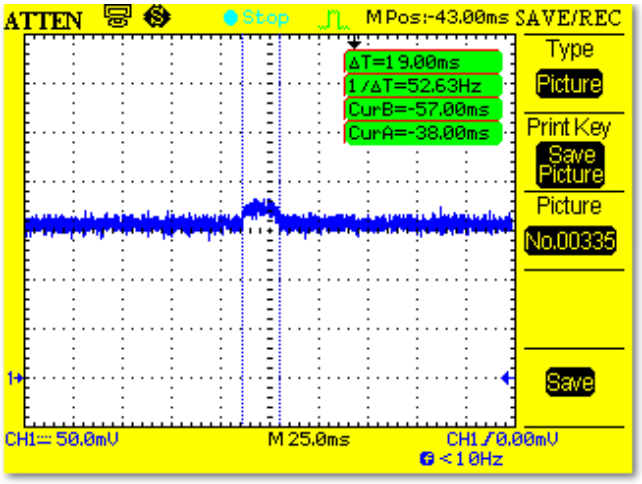


Fig. C.38.b. Tiempo de ejecución de la interrupción

Cuando se produce una interrupción en el sensor y se cierra, se produce ese pico que se ve en las gráficas.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energetic total (mA·ms)
Lectura sensor PTFA3415	29	1.02	29.852

Tabla C.43. Consumo de la lectura del sensor PTFA3415

C.2.3.7. Sensor de nivel de líquido PTFA1103

Este ejemplo muestra como leer el sensor PTFA1103 en el socket 1 de la placa de Eventos. Para usar este ejemplo, se debe activar el primer interruptor. Es necesario conocer que el consume en el estado on de la placa activando este interruptor es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

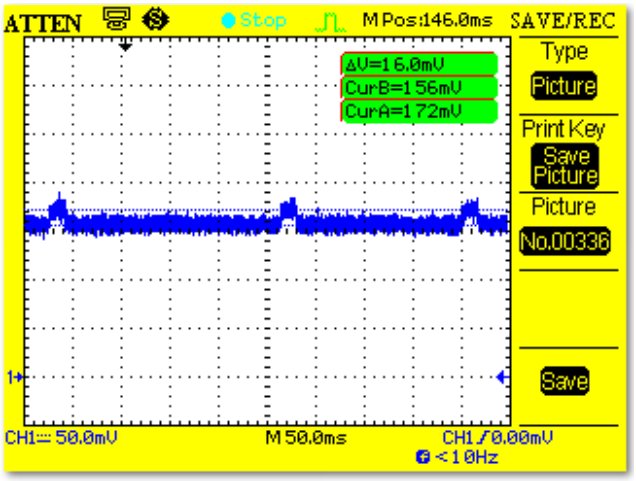


Fig. C.39.a. Medida general de lectura del sensor

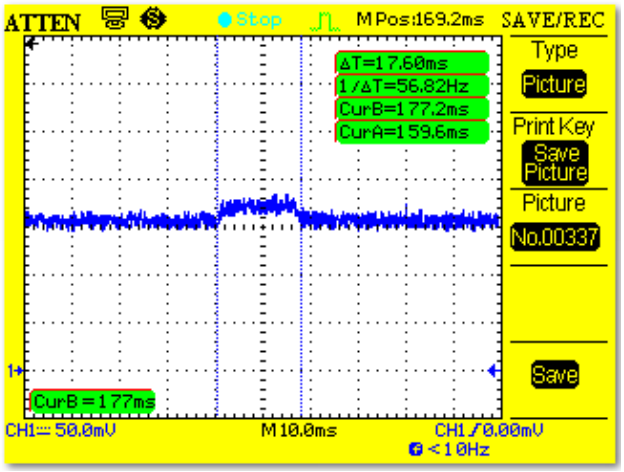


Fig. C.39.b. Tiempo de ejecución de la interrupción

Cuando se produce una interrupción en el sensor y se cierra, se produce ese pico que se ve en las gráficas.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor PTFA1103	27.6	1	27.65

Tabla C.44. Consumo de la lectura del sensor PTFA1103

C.2.3.8. Sensor de nivel de líquido PTFA0100

Este ejemplo muestra como leer el sensor PTFA0100 en el socket 1 de la placa de Eventos. Para usar este ejemplo, se debe activar el primer interruptor. Es necesario saber que el consumo en el estado on de la placa activando este interruptor es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

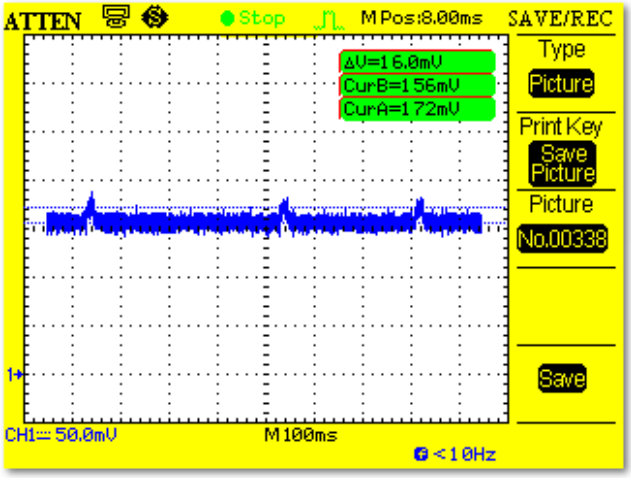


Fig. C.40.a. Medida general de lectura del sensor

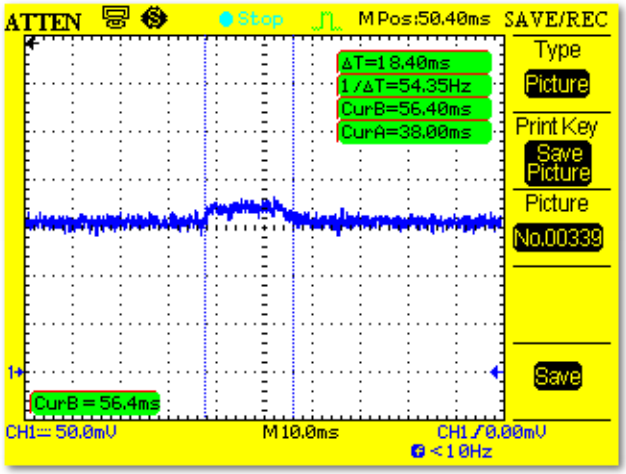


Fig. C.40.b. Tiempo de ejecución de la interrupción

Cuando se produce una interrupción en el sensor y se cierra, se produce ese pico que se ve en las gráficas.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor PTFA0100	28.4	1.01	28.91

Tabla C.45. Consumo de la lectura del sensor PTFA0100

C.2.3.9. Sensor de luminosidad (LDR)

Este ejemplo muestra como leer el sensor LDR en el socket 3 de la placa de Eventos. Para usar este ejemplo, se debe de activar el tercer interruptor. Es necesario saber que el consume en el estado on de la placa activando este interruptor, es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

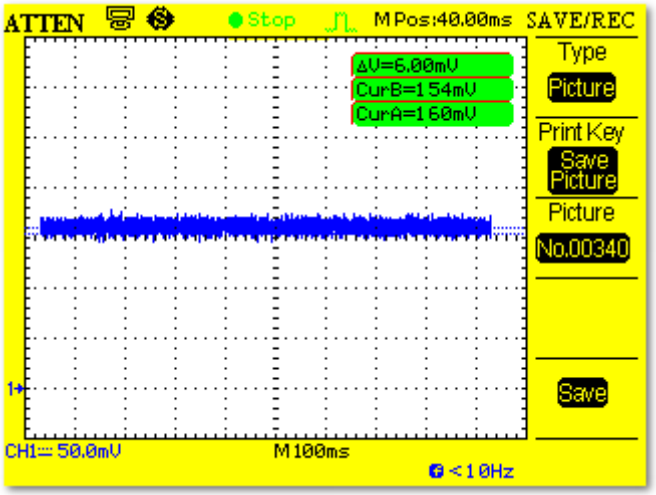


Fig. C.41. Medida general de lectura del sensor

La medida del consumo del sensor se ha hecho desde el socket 3 de la placa porque el consumo es más alto que en el socket 1 o 2, con lo cual, se cogió el caso más desfavorable, es decir el que consume más.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor LDR	10	0.368	3.68

Tabla C.46. Consumo de la lectura del sensor LDR

C.2.3.10. Sensor de presencia (PIR)

Este ejemplo muestra como leer el sensor PIR en el socket 7 de la placa de Eventos. Para usar este ejemplo, se debe activar el séptimo interruptor. Es necesario conocer que el consumo en el estado on de la placa activando este interruptor es de 0 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

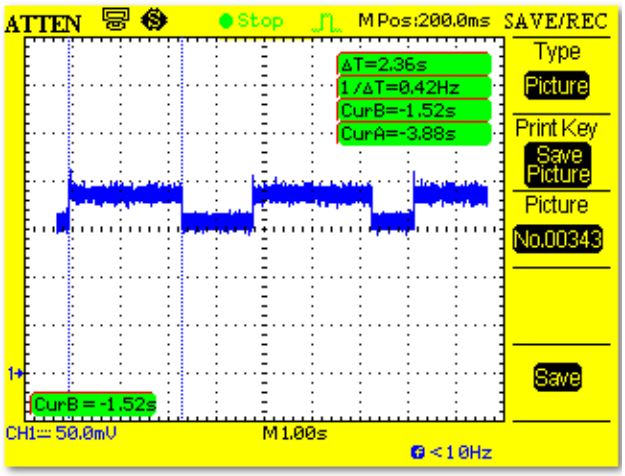


Fig. C.42.a. Tiempo de interrupción del sensor

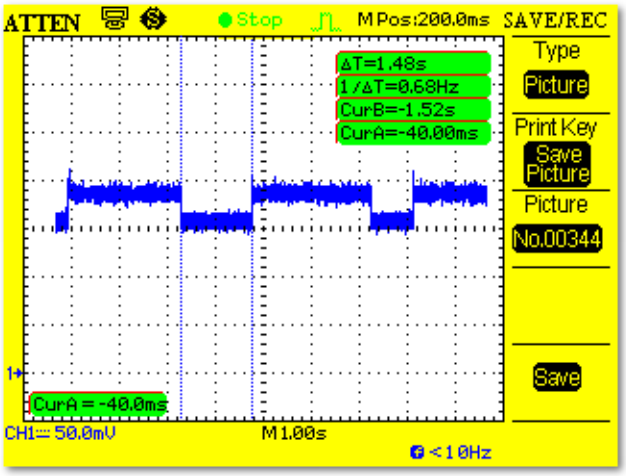


Fig. C.42.b. Tiempo en off del sensor

Cuando se produce una interrupción en el sensor y detecta una presencia, se produce ese consumo extra que se ve en las gráficas.

El consumo del sensor varía de acuerdo al tiempo en el que está detectando esa presencia.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor PIR	2360	3	7080

Tabla C.47. Consumo de la lectura del sensor PIR

C.2.3.11. Sensor Efecto Hall

Este ejemplo muestra como leer el sensor efecto Hall en el socket 1 de la placa de Eventos. Para usar este ejemplo, se debe activar el primer interruptor. Es necesario saber que el consumo en el estado on de la placa activando este interruptor, es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

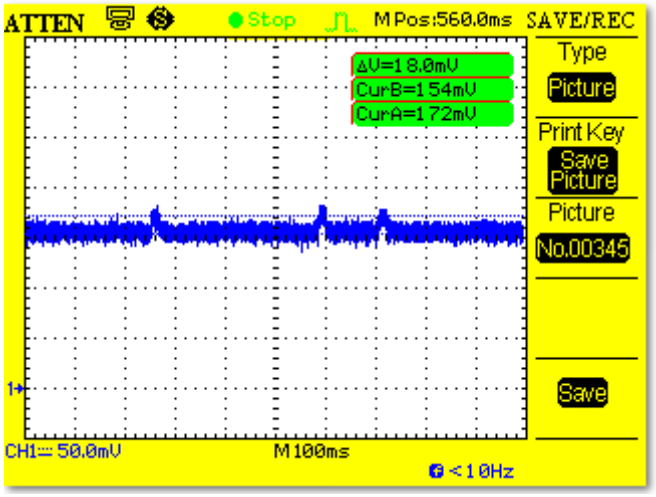


Fig. C.43. Medida general lectura del sensor

Cuando se produce una interrupción en el sensor se produce ese consumo extra que se ve en las gráficas.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor efecto Hall	26	0.96	25.133

Tabla C.48. Consumo de la lectura del sensor efecto Hall

C.2.3.12. Sensor detección de líquido (Point)

Este ejemplo muestra como leer el sensor de detección de líquido en el socket 1 de la placa de Eventos. Para usar este ejemplo, se debe activar el primer interruptor. Es necesario conocer que el consumo en el estado on de la placa activando este interruptor es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

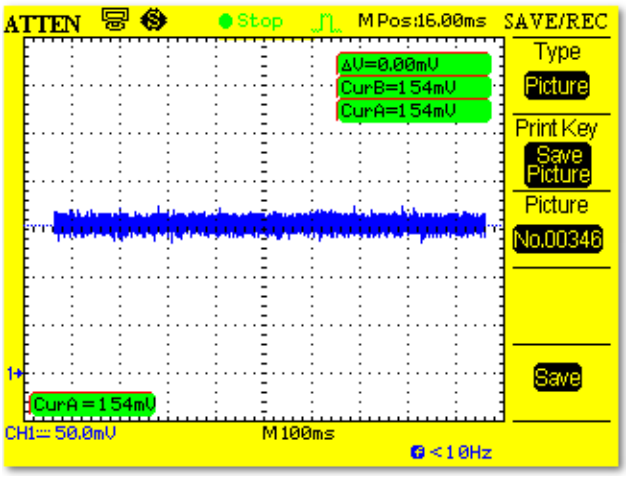


Fig. C.44. Medida general de lectura del sensor

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor detección líquido	10	0.005	0.05

Tabla C.49. Consumo de la lectura del sensor de detección de líquido

C.2.3.13. Sensor de elongación

Este ejemplo muestra como leer el sensor de elongación en el socket 3 de la placa de Eventos. Para usar este ejemplo, se debe activar el tercer interruptor. Es necesario saber que el consumo en el estado on de la placa activando este interruptor es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

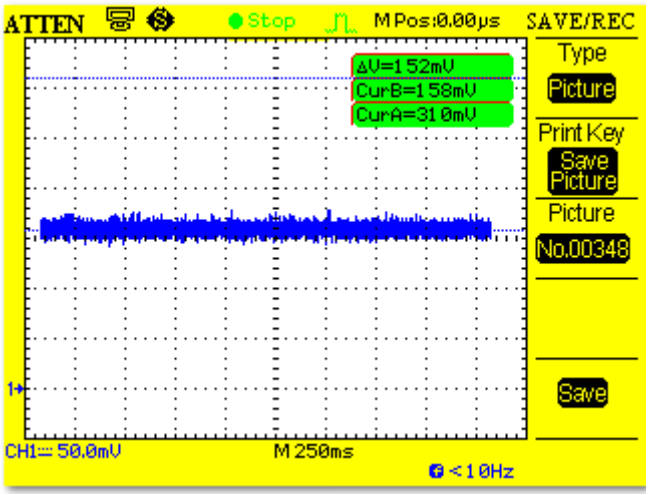


Fig. C.45. Medida general de lectura del sensor

La medida del consumo del sensor se ha hecho desde el socket 3 de la placa porque el consumo es más alto que en el socket 1 o 2, con lo cual, se cogió el caso más desfavorable, es decir el que consume más.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor elongación	10	0.28	2.8

Tabla C.50. Consumo de la lectura del sensor de elongación

C.2.3.14. Sensor de humedad

Este ejemplo muestra como leer el sensor de humedad en el socket 6 de la placa de Eventos. Para usar este ejemplo, se debe activar el sexto interruptor. ES necesario conocer que el consume en el estado on de la placa activando este interruptor es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

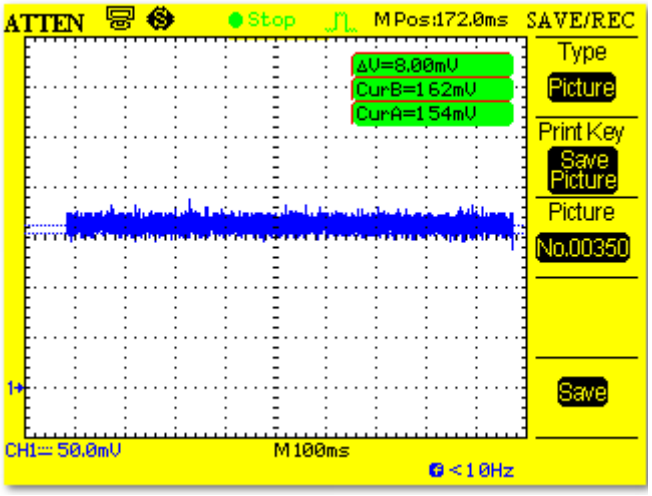


Fig. C.46. Medida general de lectura del sensor

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor humedad	10	0.568	5.68

Tabla C.51. Consumo de la lectura del sensor de humedad

C.2.3.15. Sensor de detección de líquido (Line)

Este ejemplo muestra como leer el sensor de detección de líquido en el socket 1 de la placa de Eventos. Para usar este ejemplo, se deber activar el primer interruptor. Es necesario saber que el consumo en el estado on de la placa activando este interruptor es 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

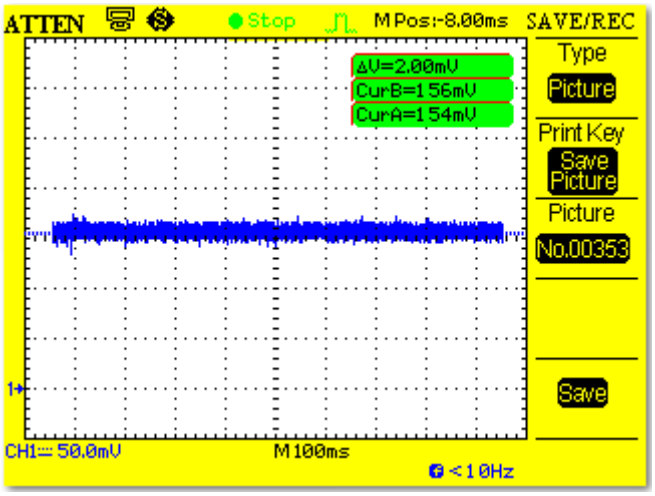


Fig. C.47. Medida general de lectura del sensor

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor detección líquido	10	0.005	0.05

Tabla C.52. Consumo de la lectura del sensor de detección de líquido

C.2.3.16. Sensor de flujo de líquido FS100A Y FS200B

Este ejemplo muestra como leer los sensores FS100A Y FS200B de la placa de Eventos. Este sensor se debe conectar en el socket 8 colocando el jumper en la posición pull up. Es necesario saber que el consumo en el estado on de la placa activando este interruptor es de 32 uA.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

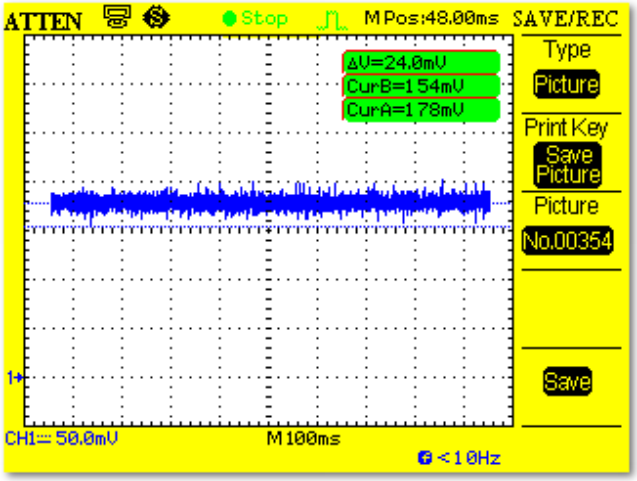


Fig. C.48.a. Medida general de lectura del sensor

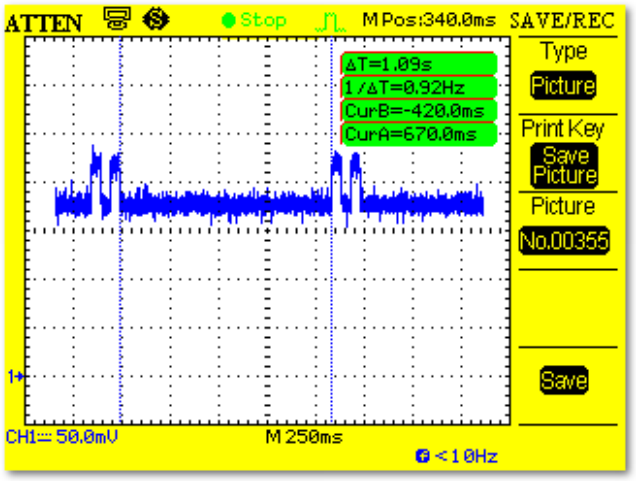


Fig. C.48.b. Tiempo de ejecución de la lectura del sensor

Para distinguir en la gráfica el tiempo de ejecución de la lectura del sensor, se añade al código una serie de funciones que producen un consumo extra a lo medido y así distinguirlo con facilidad, por lo tanto, la lectura del sensor le cuesta ejecturase 1040 ms.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetic total (mA·ms)**
Lectura sensor flujo líquido	1040	2.168	2254.72

Tabla C.53. Consumo de la lectura del sensor de flujo de líquido

C.2.4. Placa de Gases

C.2.4.1. Proceso de encendido y apagado de la placa de Gases

Este ejemplo muestra como encender y apagar la placa de Gases cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

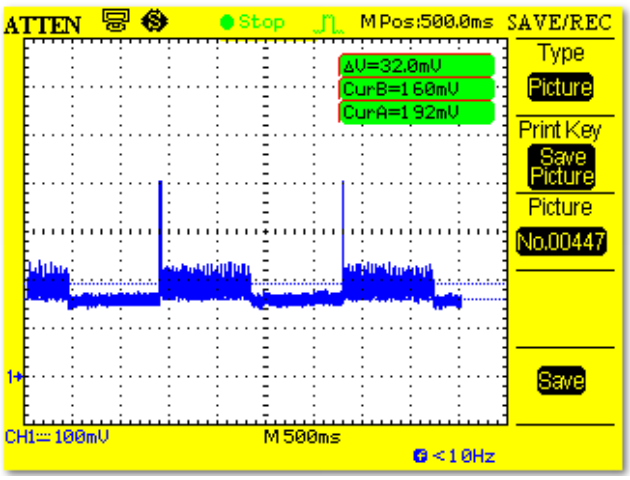


Fig. C.49.a. Medida general del proceso

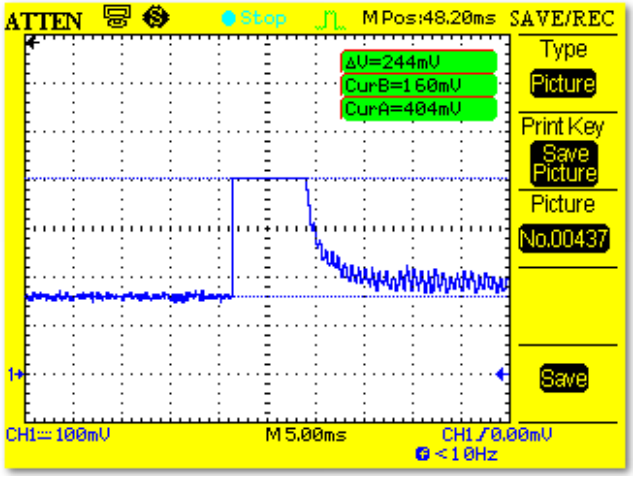


Fig. C.49.b. Pico de carga al encender la placa

Como se ve en la gráfica (a), se produce un consumo extra de 600 uA en el estado off de la placa de Gases. Es debido al diseño hardware de la placa.

Para el cálculo del consumo, se ha tomado como referencia el estado on del Waspmote, es decir los 15.6 mA.

El consumo del proceso y estado on y proceso off de la placa se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso on	8	24.8	198.4
Estado on	1000	3.7	3769.6
Proceso off	0	3.7	0

Tabla C.54. Consumo del proceso y estado on y proceso off de la placa de Gases

C.2.4.2. Sensor de humedad

Este ejemplo muestra como leer el sensor de humedad de la placa de Gases cada 15 segundos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

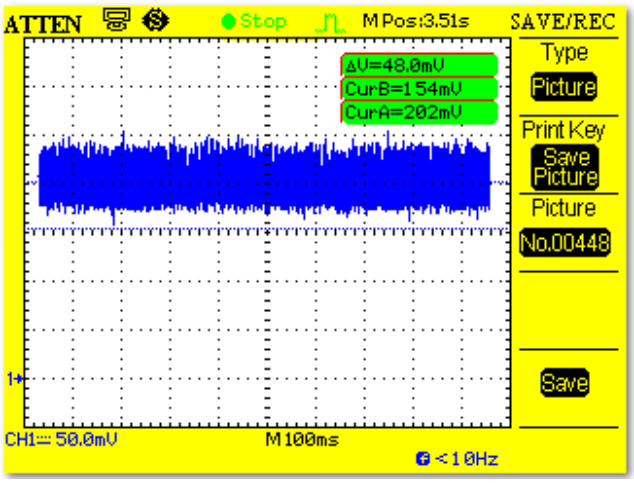


Fig. C.50.a. Medida general del proceso

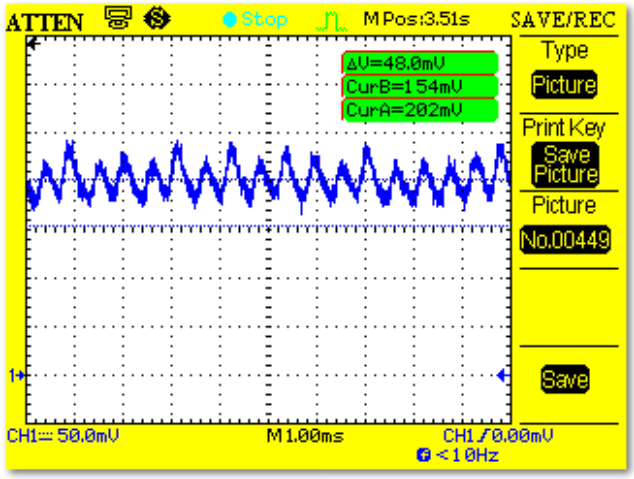


Fig. C.50.b. Proceso de lectura del sensor

El consumo de lectura del sensor de humedad de la placa se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Lectura sensor humedad	15000	0.9	13500

Tabla C.55. Consumo de la lectura del sensor de humedad

C.2.4.3. Sensor de temperatura

Este ejemplo muestra como leer el sensor de temperatura de la placa de Gases.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

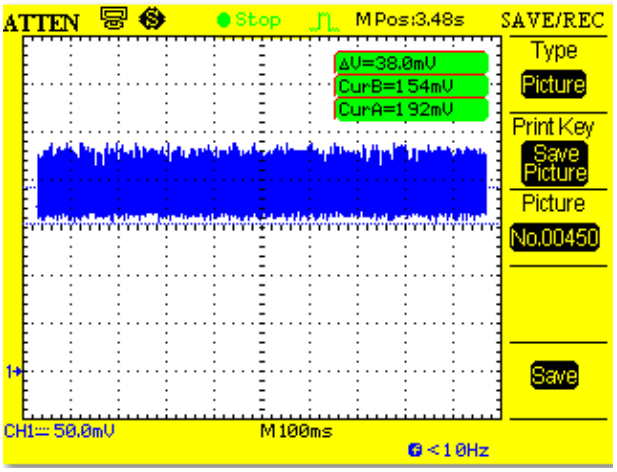


Fig. C.51.a. Medida general del proceso

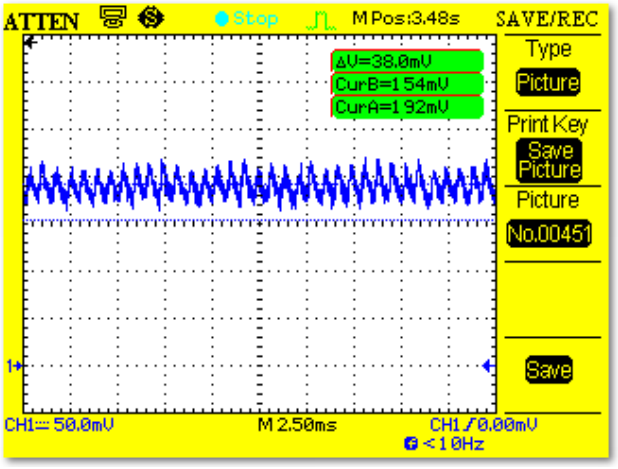


Fig. C.51.b. Proceso de lectura del sensor

El consumo de lectura del sensor de temperatura de la placa se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Lectura sensor temperatura	100	0.005	0.5

Tabla C.56. Consumo de la lectura del sensor de humedad

C.2.4.4. Sensor CO

Este ejemplo muestra como leer el sensor CO cada segundo colocado en el socket 4 de la placa de Gases.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

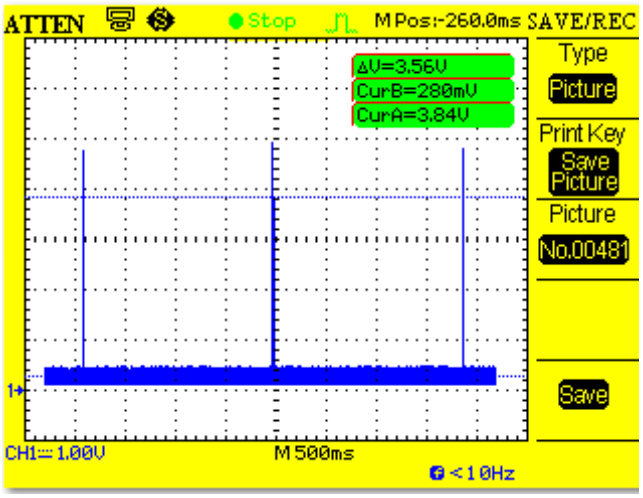


Fig. C.52.a. Medida general del proceso

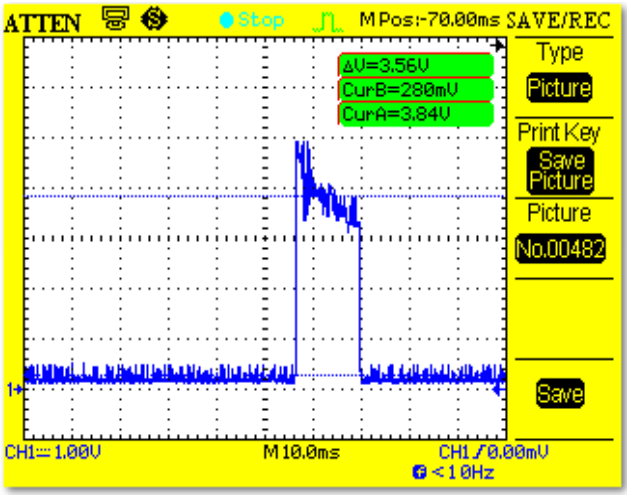


Fig. C.52.b. Pico de lectura del sensor

La gráfica sale así porque la función de lectura del sensor hace una llamada a la función pulse, que genera dos pulsos en DIG5 Y DIG6 con un delay entre pulsos de 980 ms, por lo que es normal que dure un segundo.

El consumo de lectura del sensor CO de la placa se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Lectura sensor CO	993.2	5.23	5232

Tabla C.57. Consumo de la lectura del sensor CO

C.2.4.5. Sensor CO_2

Este ejemplo muestra como leer el sensor CO_2 de la placa de Gases cada segundo. Para hacer esto, es necesario encender el sensor previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

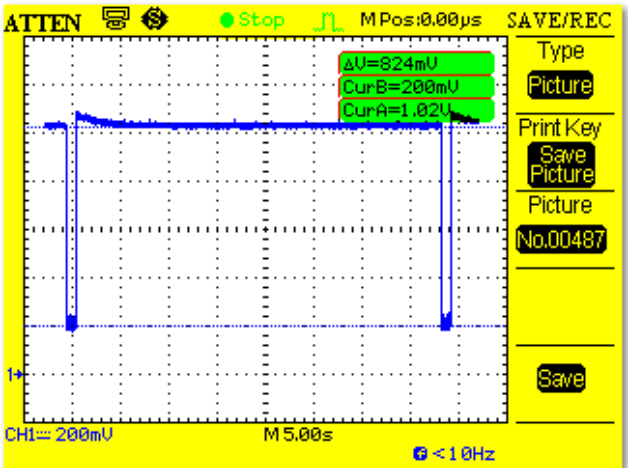


Fig. C.53.a. Consumo proceso total

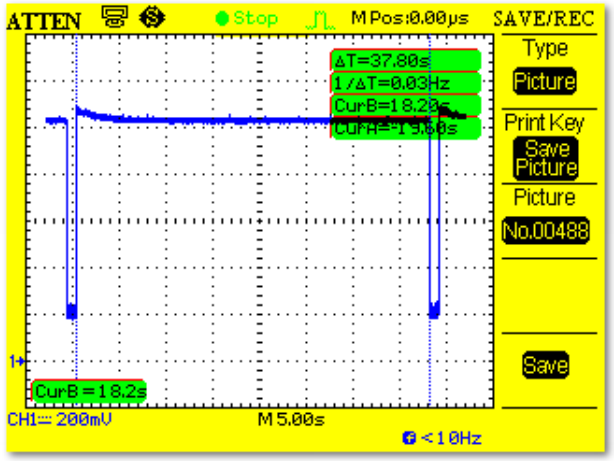


Fig. C.53.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor CO_2 de la placa se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	40000	82.8	3312000

Tabla C.58. Consumo del proceso total para la lectura del sensor CO_2

C.2.4.6. Sensor O_2

Este ejemplo muestra como leer el sensor O_2 de la placa de Gases.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

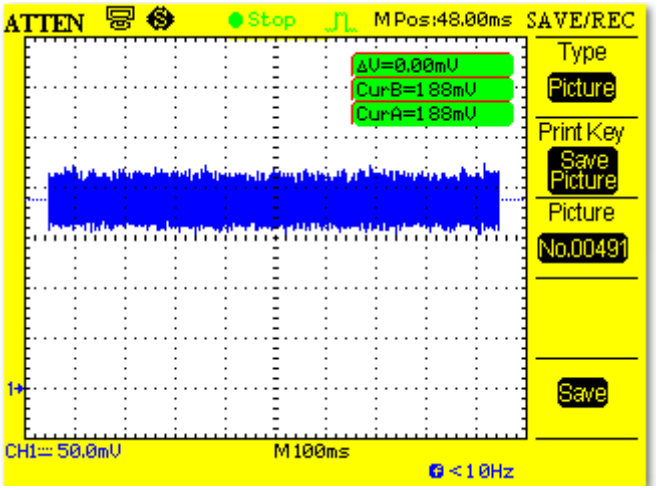


Fig. C.54. Proceso lectura sensor

El consumo de lectura del sensor O_2 de la placa se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Lectura sensor O_2	10	0	0

Tabla C.59. Consumo de la lectura del sensor O_2

Se observa que el consumo de la lectura del sensor O_2 es 0 ms.

C.2.4.7. Sensor NO_2

Este ejemplo muestra como leer el sensor NO_2 de la placa de Gases cada segundo. Para hacer esto, el sensor se tiene que encenderse en el socket 3B de la placa previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

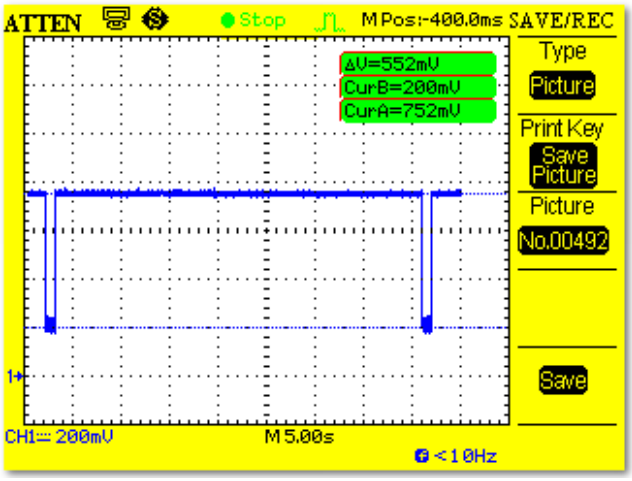


Fig. C.55.a. Consumo proceso total

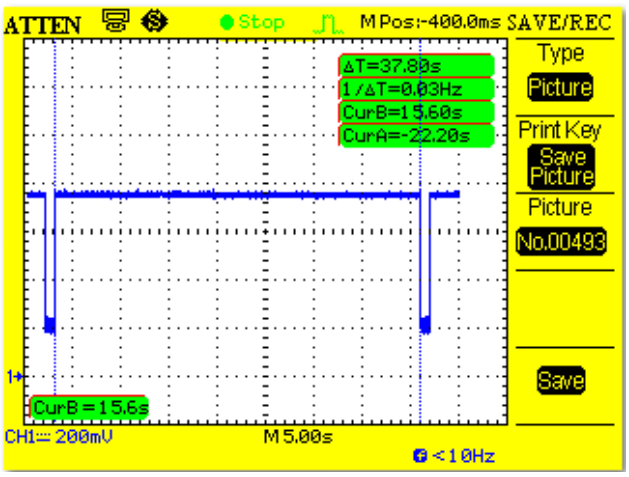


Fig. C.55.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor NO_2 de la placa se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	40000	55.9	2236000

Tabla C.60. Consumo del proceso total para la lectura del sensor NO_2

C.2.4.8. Sensor NH_3

Este ejemplo muestra como leer el sensor NH_3 de la placa de Gases en el socket 3.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

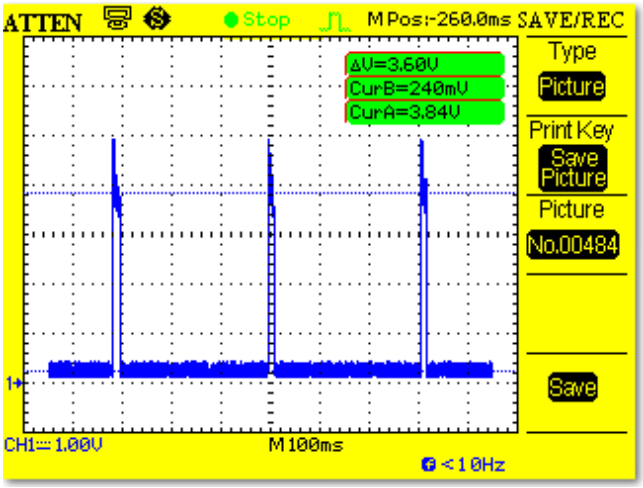


Fig. C.56.a. Medida general del proceso

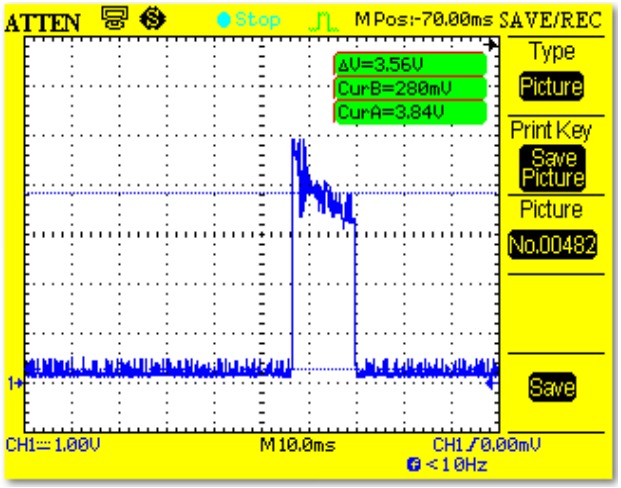


Fig. C.56.b. Pico de lectura del sensor

La gráfica sale así porque la función de lectura del sensor hace una llamada a la función pulse, que genera dos pulsos en DIG5 Y DIG6 con un delay entre pulsos de 300 ms, por lo que es normal que dure todo el proceso de lectura del sensor eso.

El consumo de lectura del sensor NH_3 de la placa se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Lectura sensor NH_3	313.2	16.7	5232

Tabla C.61. Consumo del proceso total para la lectura del sensor NH_3

C.2.4.9. Sensor CH_4

Este ejemplo muestra como leer el sensor CH_4 de la placa de Gases cada segundo. Para hacer esto, el sensor se tiene que encenderse en el socket 2A de la placa.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

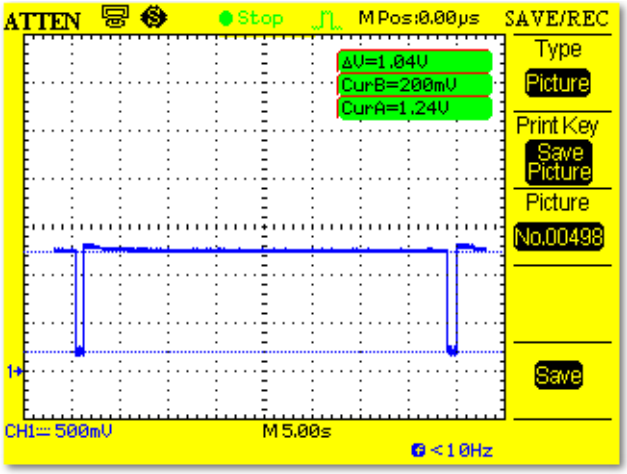


Fig. C.57.a. Consumo proceso total

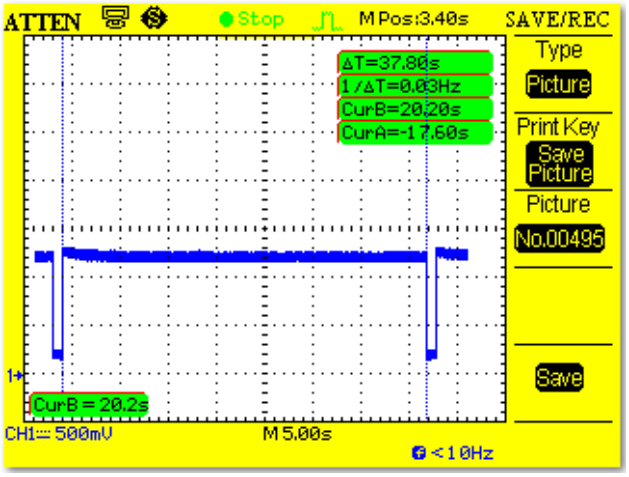


Fig. C.57.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	40000	104.7	4188000

Tabla C.62. Consumo del proceso total para la lectura del sensor CH_4

C.2.4.10. Sensor TGS2610

Este ejemplo muestra como leer el sensor TGS2610 de la placa de Gases cada segundo. Para hacer esto, el sensor se tiene que encenderse en el socket 2A de la placa.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

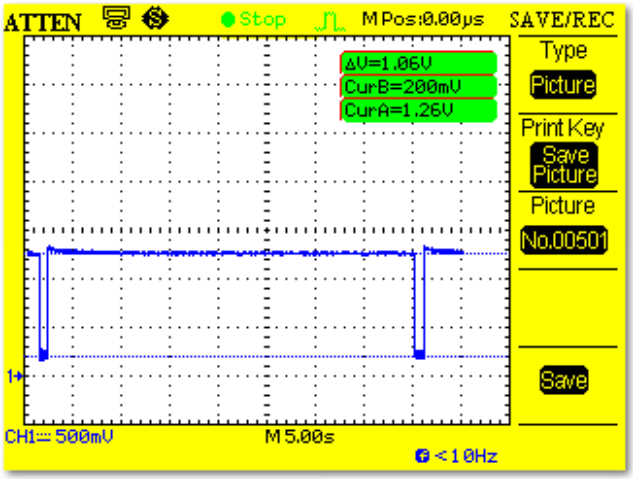


Fig. C.58.a. Consumo proceso total

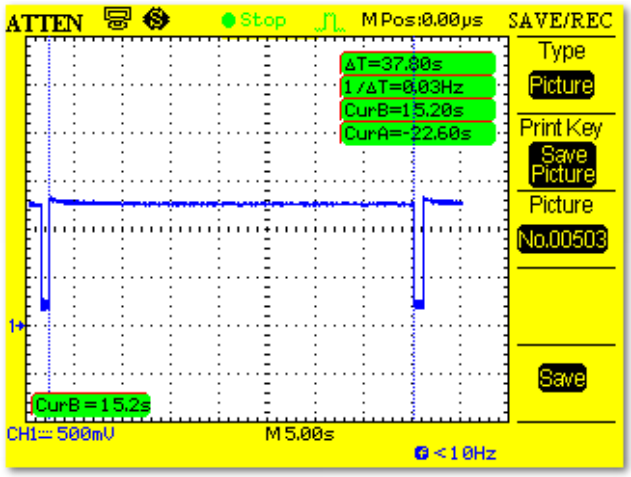


Fig. C.58.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	40000	106.7	4268000

Tabla C.63. Consumo del proceso total para la lectura del sensor TGS2610

C.2.4.11. Sensor TGS2600

Este ejemplo muestra como leer el sensor TGS2600 de la placa de Gases. Para hacer esto, el sensor se tiene que encenderse en el socket 2A previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

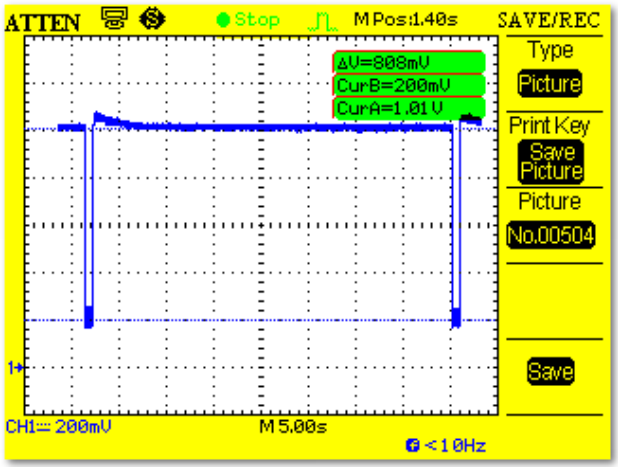


Fig. C.59.a. Consumo proceso total

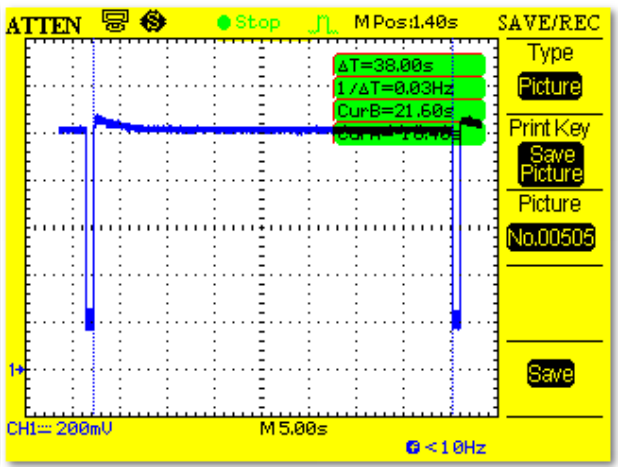


Fig. C.59.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	40000	81.7	3268000

Tabla C.64. Consumo del proceso total para la lectura del sensor TGS2600

C.2.4.12. Sensor TGS2602

Este ejemplo muestra como leer el sensor TGS2602 de la placa de Gases. Para hacer esto, el sensor se tiene que encenderse en el socket 2A de la placa.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

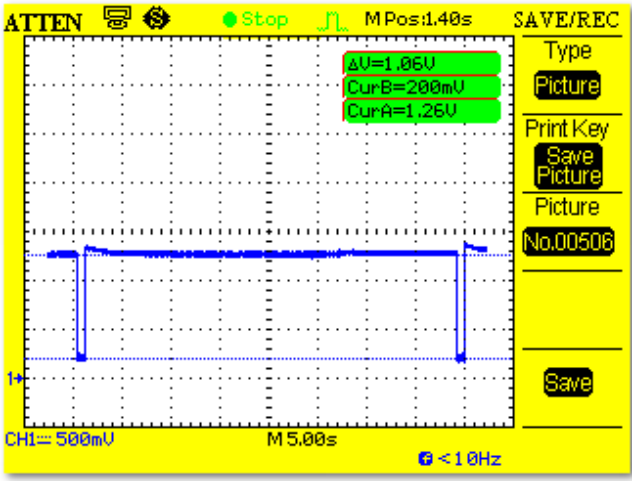


Fig. C.60.a. Consumo proceso total

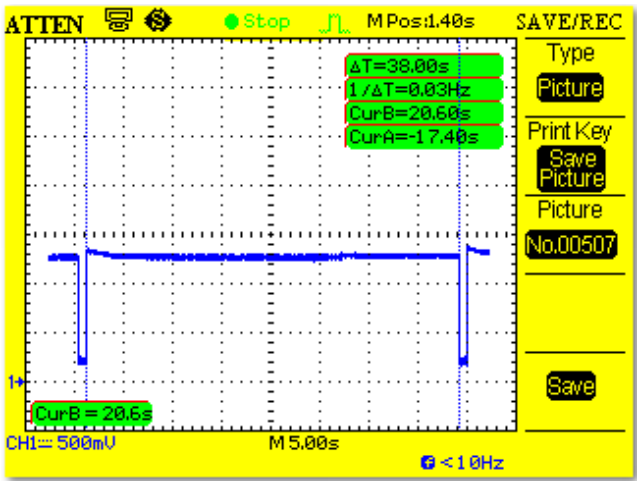


Fig. C.60.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	40000	106.7	4268000

Tabla C.65. Consumo del proceso total para la lectura del sensor TGS2602

C.2.4.13. Sensor TGS2620

Este ejemplo muestra como leer el sensor TGS2620 de la placa de Gases. Para hacer esto, el sensor se tiene que encenderse en el socket 2A de la placa.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

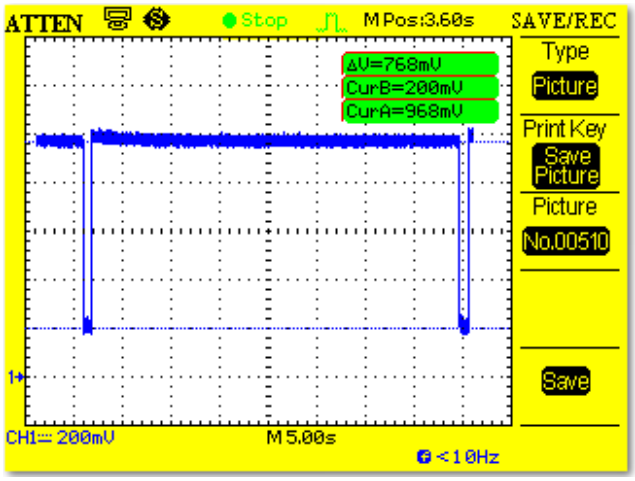


Fig. C.61.a. Consumo proceso total

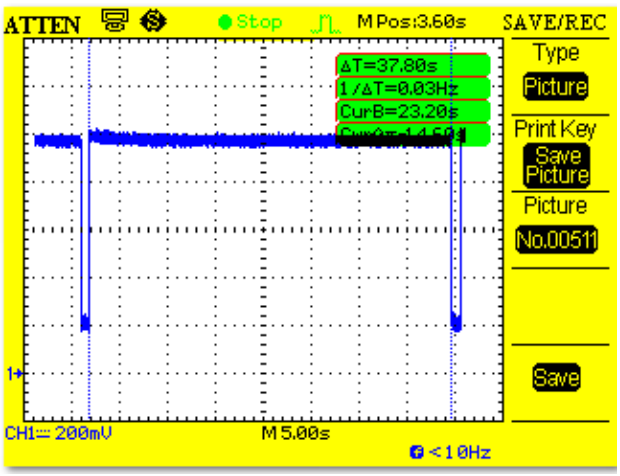


Fig. C.61.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	40000	77.5	3100000

Tabla C.66. Consumo del proceso total para la lectura del sensor TGS2620

C.2.4.14. Sensor MiCS-2610

Este ejemplo muestra como leer el sensor MiCS-2610 de la placa de Gases. Para hacer esto, el sensor se tiene que encender en el socket 2B de la placa.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

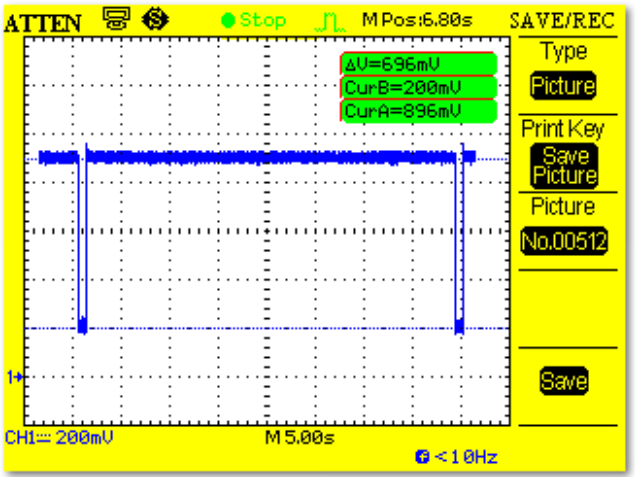


Fig. C.62.a. Consumo proceso total

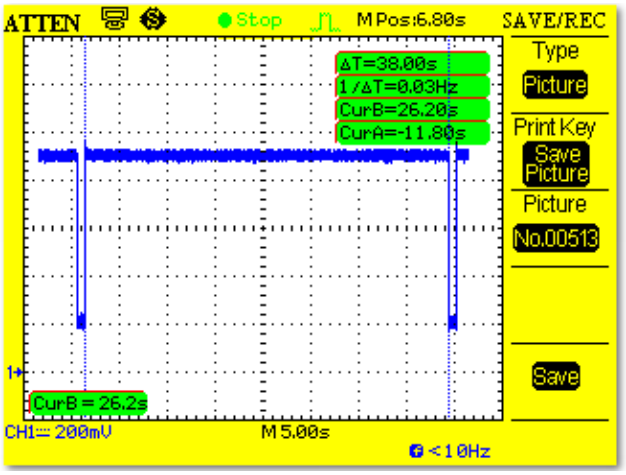


Fig. C.62.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	40000	70.3	2812000

Tabla C.67. Consumo del proceso total para la lectura del sensor MiCS-2610

C.2.4.15. Sensor MiCS-5521

Este ejemplo muestra como leer el sensor MiCS-5521 de la placa Gases. Para hacer esto, el sensor se tiene que encender en el socket 2B de la placa.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

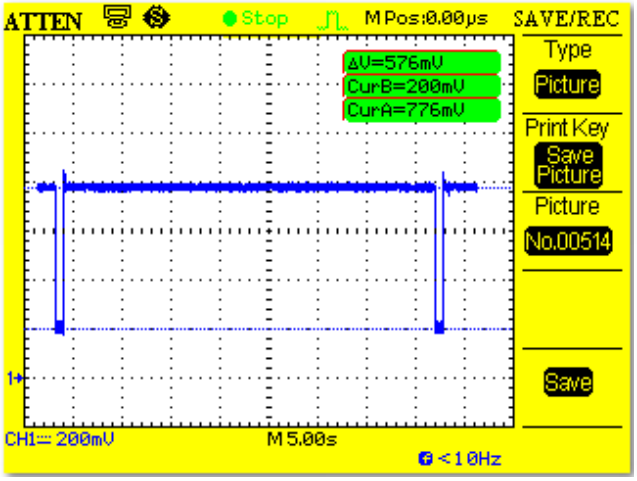


Fig. C.63.a. Consumo proceso total

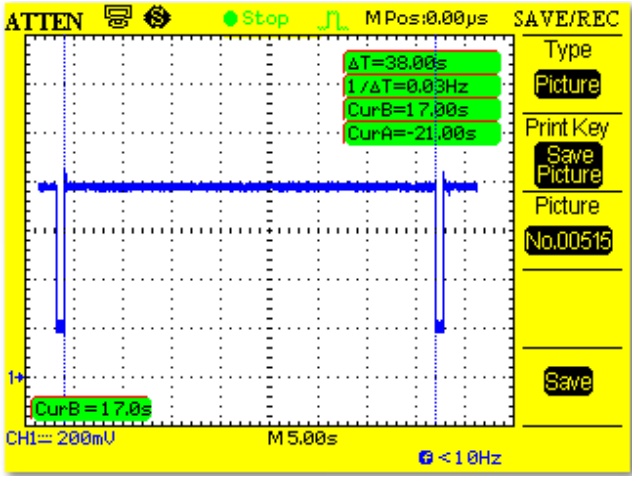


Fig. C.63.b. Tiempo de ejecución del proceso

El consumo del proceso total para la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	40000	58.3	2332000

Tabla C.68. Consumo del proceso total para la lectura del sensor MiCS-5521

C.2.5. Placa de Smart Parking

Antes de empezar a ejecutar cualquier acción que realiza esta placa, lo primero de todo se debe de cargar un código en Waspmote que sirve para compensar la variación de temperatura producida en el sensor de temperatura de la placa. Cada placa de Parking, tiene unos coeficientes particulares usados en la compensación de la temperatura. Se encargarán de guardarse en la EEPROM del microcontrolador de Waspmote entre las direcciones 186 y 222. (El código se encuentra en el apartado de la placa de Parking de los Anexos del documento en word de la Memoria en el CD adjunto).

C.2.5.1. Proceso de encendido y apagado de la placa de Parking

Este ejemplo muestra como encender y apagar la placa de Parking cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

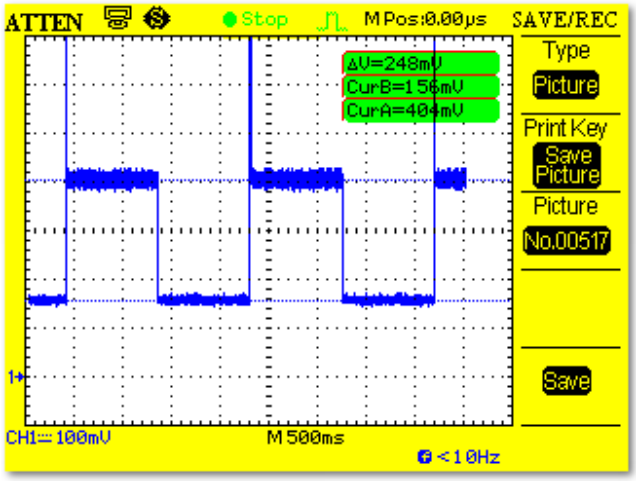


Fig. C.64.a. Medida general del proceso

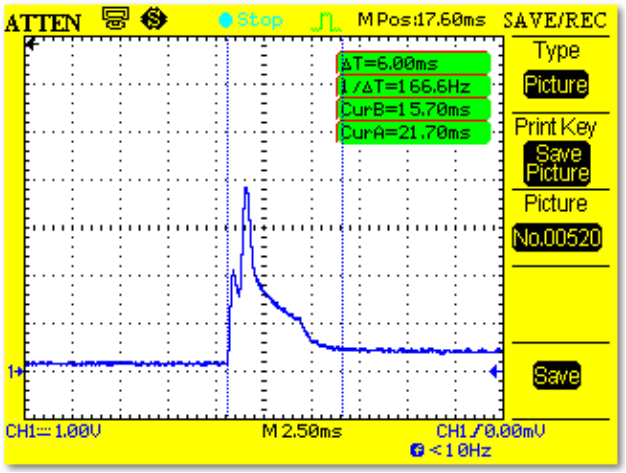


Fig. C.64.b. Pico de carga del encendido de la placa

El consumo del proceso y estado on y proceso off de la placa se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso on	6	114.208	685.25
Estado on	1000	24.8	24800
Proceso off	0	24.8	0

Tabla C.69. Consumo del proceso y estado on y proceso off de la placa de Parking

C.2.5.2. Lectura del sensor MFS

Este ejemplo muestra como leer el sensor MFS de la placa de Parking.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

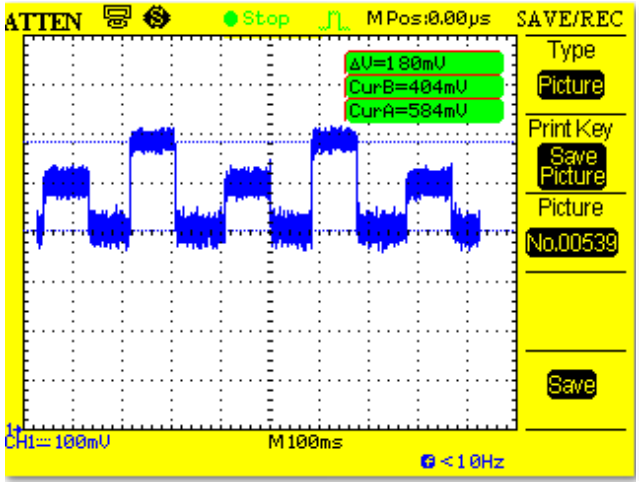


Fig. C.65.a. Medida general del proceso

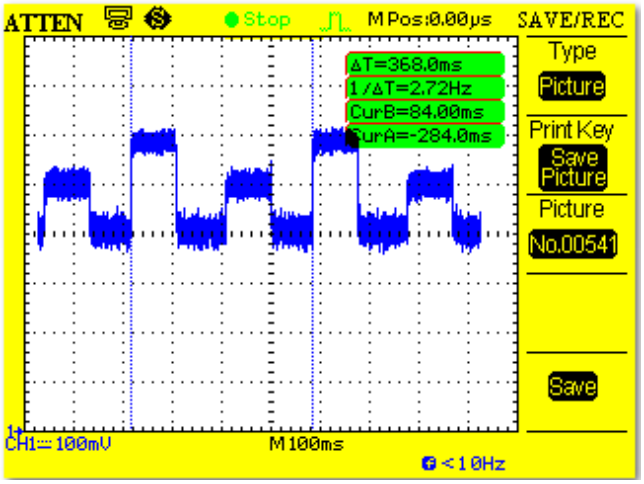


Fig. C.65.b. Tiempo de ejecución del proceso

Como se ve en las gráficas, se representan la salida de uno de los ejes del MFS.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	368	6.8	2502.4

Tabla C.70. Consumo del proceso total de lectura del MFS

C.2.6. Placa de Smart Metering

C.2.6.1. Proceso de encendido y apagado de la placa de Smart Metering y de la placa Smart Metering versión PRO

Este ejemplo muestra como encender y apagar la placa normal y la placa versión PRO de Smart Metering cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

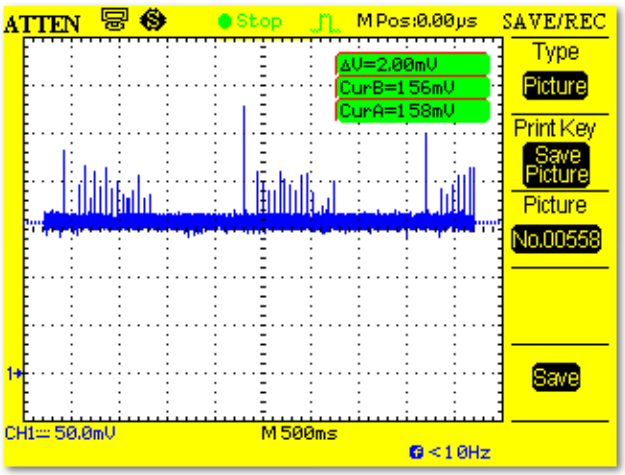


Fig. C.66.a. Medida general cada segundo

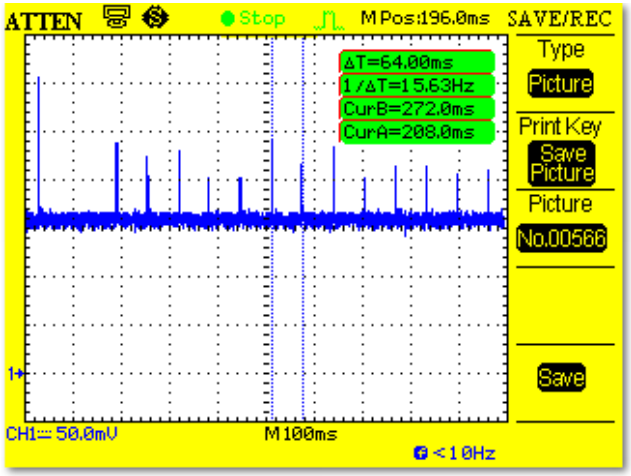


Fig. C.66.b. Tiempo entre un pico y otro

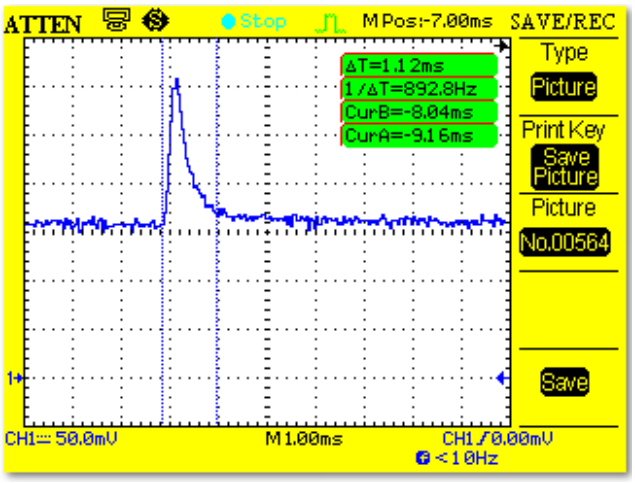


Fig. C.66.c. Anchura pulso de carga al encenderse

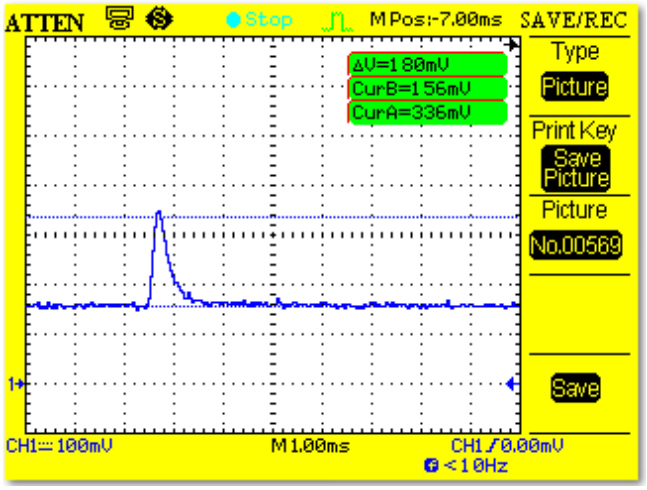


Fig. C.66.d. Altura pulso de carga

Los picos que aparecen en el estado On de la placa son debidos al conversor dc-dc. El gordo del principio es debido a la carga de condensadores, y luego aparecen picos, periódicos, con mayor frecuencia cuanto mayor sea la corriente de salida del conversor.

El consumo del proceso y estado on y proceso off de la placa se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso on	1.12	12.6	14.11
Estado on	1000	0.26	260
Proceso off	0	0.26	0

Tabla C.71. Consumo del proceso y estado on y proceso off de la placa de Smart Metering

C.2.6.2. Sensor de corriente

Este ejemplo muestra como leer el sensor de corriente cada segundo. Para hacer esto, el sensor tiene que encenderse previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

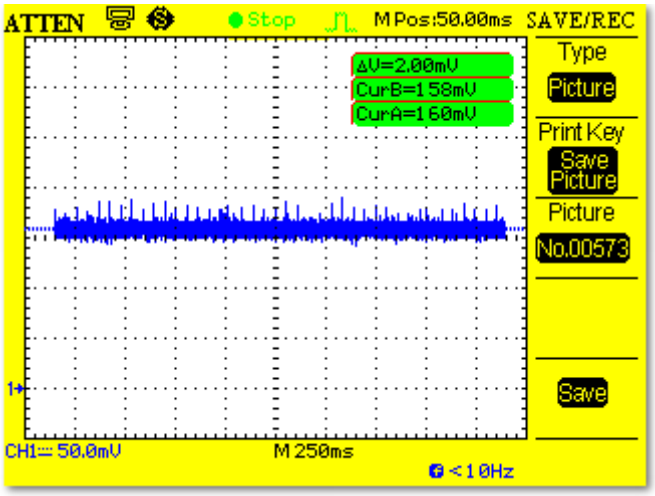


Fig. C.67. Medida general del proceso cada segundo

El consumo del proceso total para la lectura del sensor de corriente se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	50	0.2	10

Tabla C.72. Consumo del proceso total para la lectura del sensor de corriente

C.2.6.3. Célula de carga alimentado a 5V

Este ejemplo muestra como leer el sensor de célula de carga conectado en el socket de 5V de la placa de Smart Metering. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

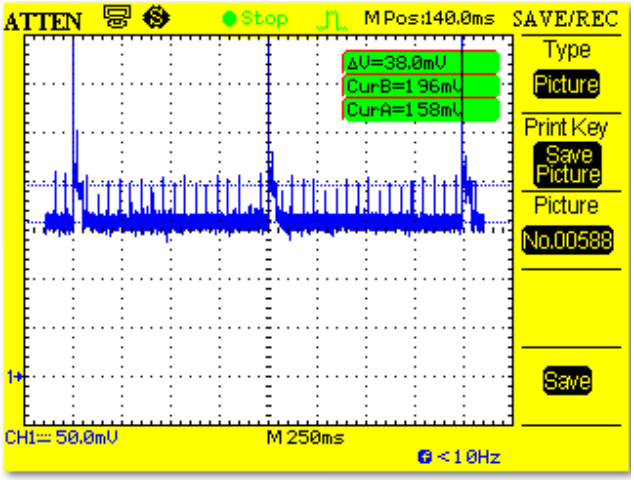


Fig. C.68.a. Medida general cada segundo

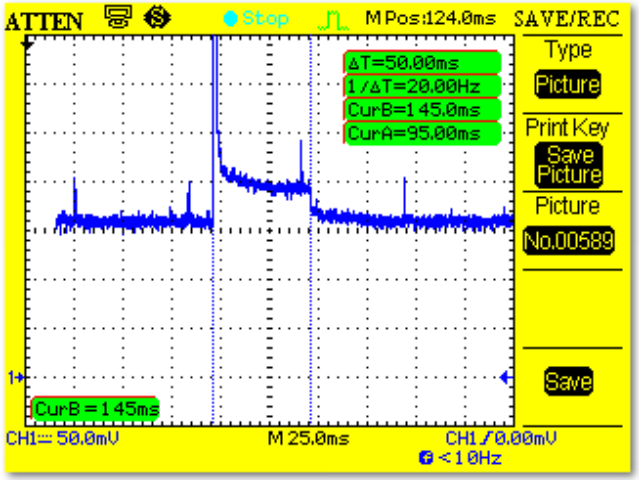


Fig. C.68.b. Anchura del proceso de lectura

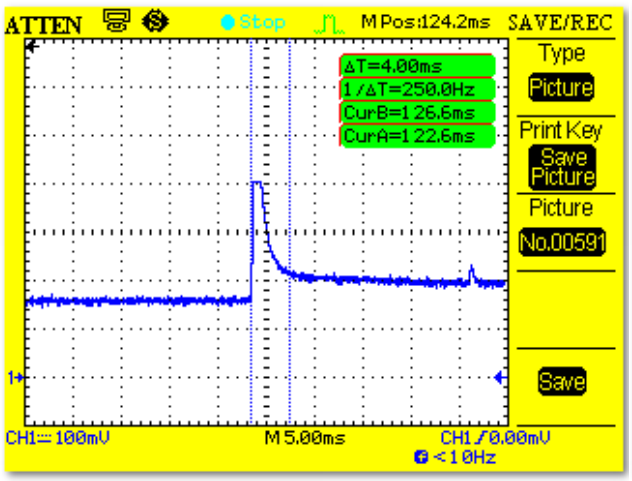


Fig. C.68.c. Anchura pulso encendido del sensor

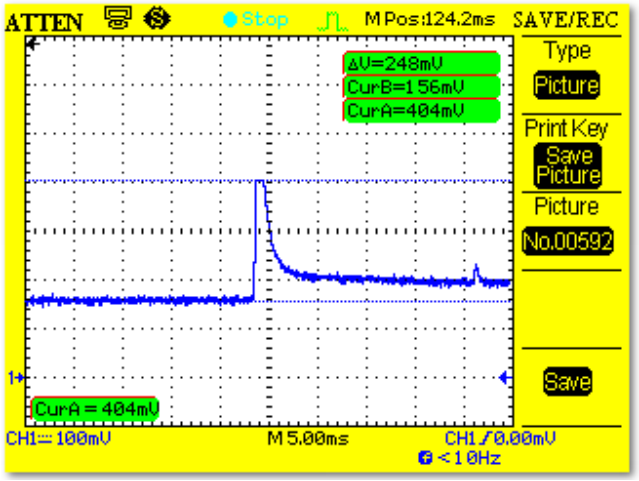


Fig. C.68.d. Altura pulso encendido del sensor

El consumo del proceso total para la lectura del sensor de célula de carga se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	50	5.05	252.88

Tabla C.73. Consumo del proceso total para la lectura del sensor de célula de carga

C.2.6.4. Célula de carga alimentado a 10V

Este ejemplo muestra como leer el sensor de célula de carga conectado en el socket de 10V de la placa de Smart Metering. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

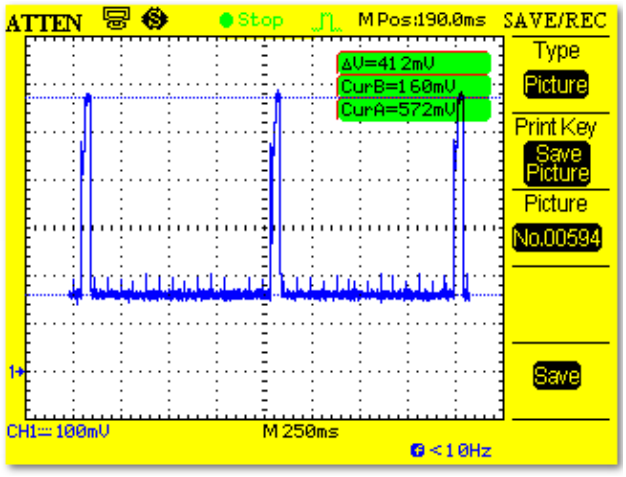


Fig. C.69.a. Medida general cada segundo

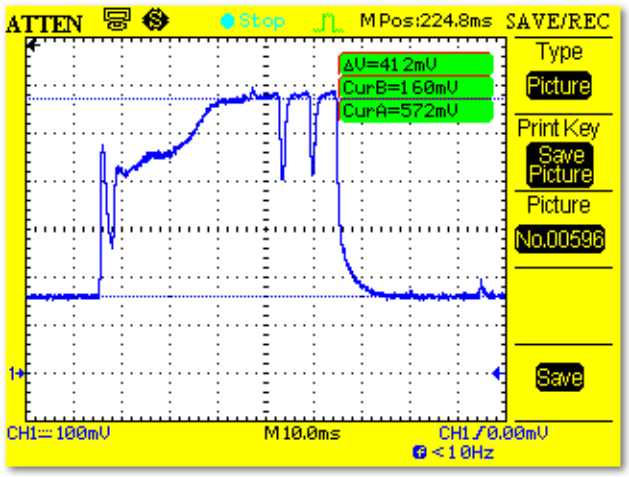


Fig. C.69.b. Altura máxima del proceso de lectura

El consumo del proceso total para la lectura del sensor de célula de carga se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	50	36.26	1813.12

Tabla C.74. Consumo del proceso total para la lectura del sensor de célula de carga

C.2.6.5. Flujo de agua alimentado a 3.3V

Este ejemplo muestra como leer el sensor de flujo de líquido conectado al socket de 3.3V de la placa de Smart Metering. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

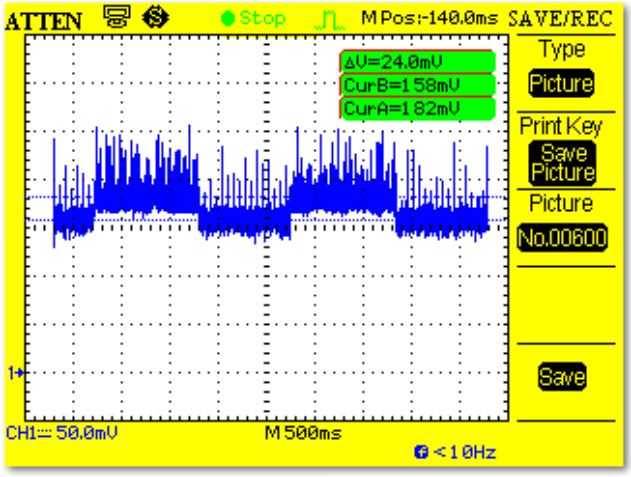


Fig. C.70.a. Medida general cada segundo

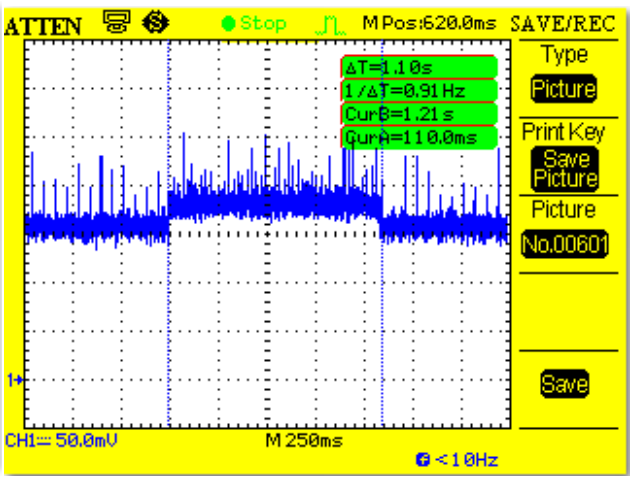


Fig. C.70.b. Anchura proceso de lectura

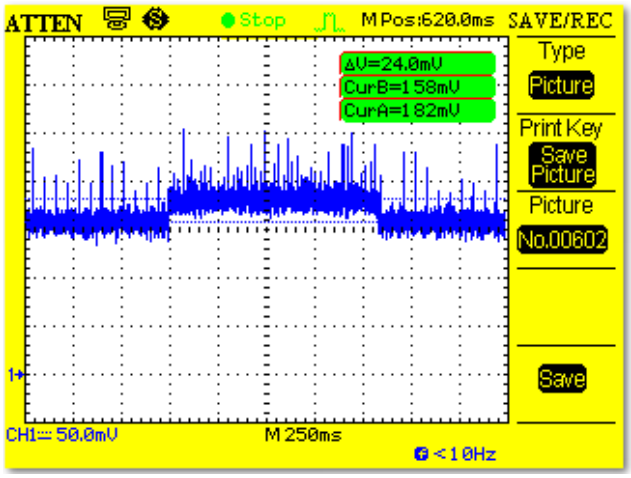


Fig. C.70.c. Altura proceso de lectura

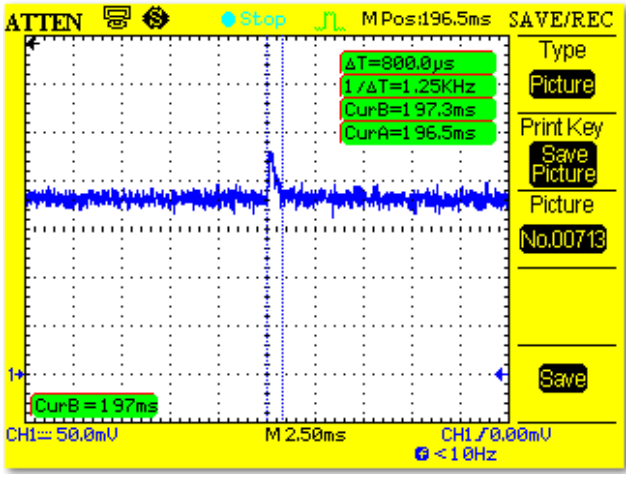


Fig. C.70.d. Anchura picos proceso de lectura

Como se ve en las imágenes, la lectura del sensor se ejecuta en 1050 ms.

El consumo del proceso total para la lectura del sensor de flujo de líquido se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	1100	2.4	2640

Tabla C.75. Consumo del proceso total para la lectura del sensor de flujo de líquido

C.2.6.6. Flujo de agua alimentado a 5V

Este ejemplo muestra como leer el sensor de flujo de líquido conectado al socket de 5V de la placa de Smart Metering. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

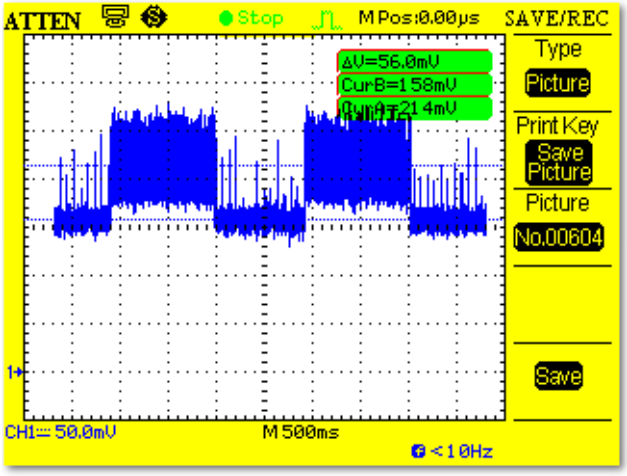


Fig. C.71.a. Medida general cada segundo

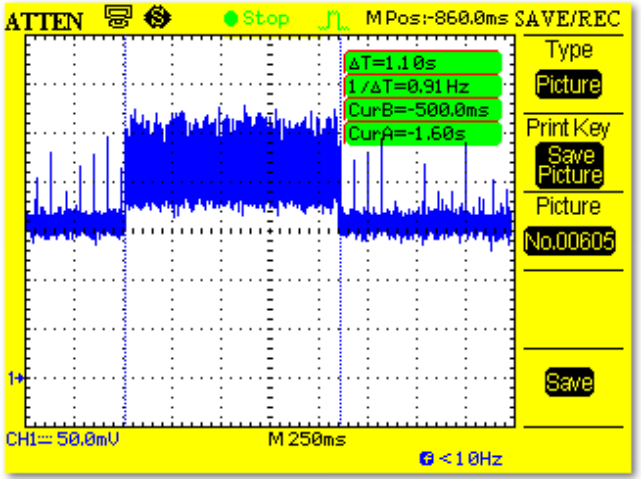


Fig. C.71.b. Anchura proceso de lectura

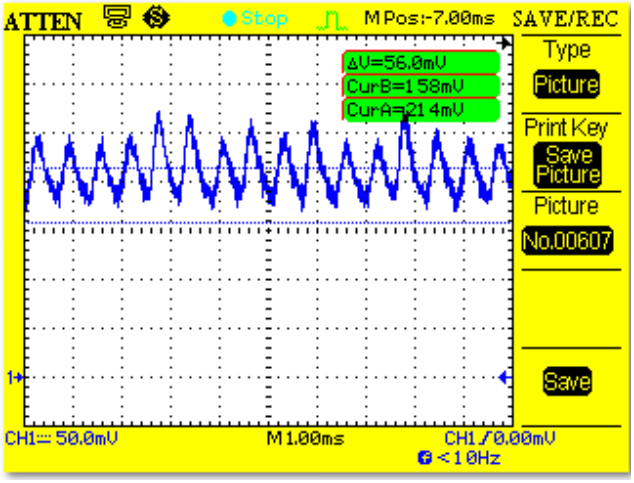


Fig. C.71.c. Altura proceso de lectura

Como se ve en las imágenes, la lectura del sensor se ejecuta en 1050 ms.

El consumo del proceso total para la lectura del sensor de flujo de líquido se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	1100	5.6	6160

Tabla C.76. Consumo del proceso total para la lectura del sensor de flujo de líquido

C.2.6.7. Sensor de humedad

Este ejemplo muestra como leer el sensor de humedad conectado a la placa de Smart Metering. Para hacer esto, el sensor tiene que encenderse previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

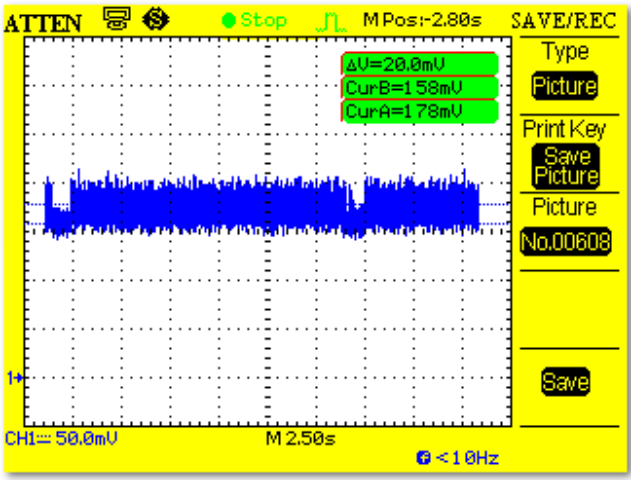


Fig. C.72.a. Medida general cada segundo

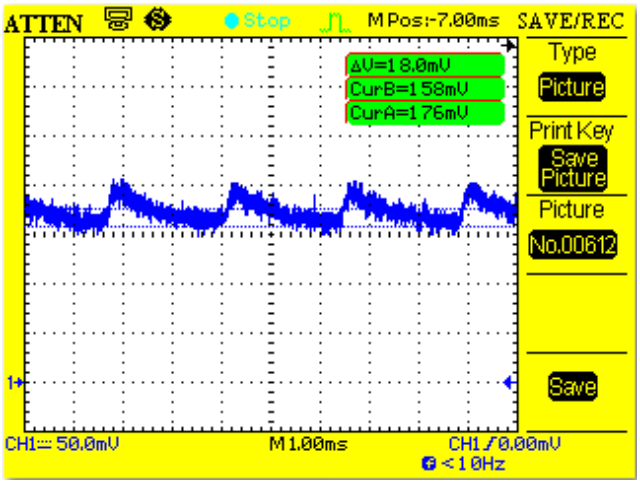


Fig. C.72.b. Altura proceso de lectura

El consumo del proceso total para la lectura del sensor de humedad se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	15000	1.8	27000

Tabla C.77. Consumo del proceso total para la lectura del sensor de humedad

C.2.6.8. Sensor de temperatura

Este ejemplo muestra como leer el sensor de temperatura conectado en la placa de Smart Metering. Para hacer esto, el sensor tiene que encenderse previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

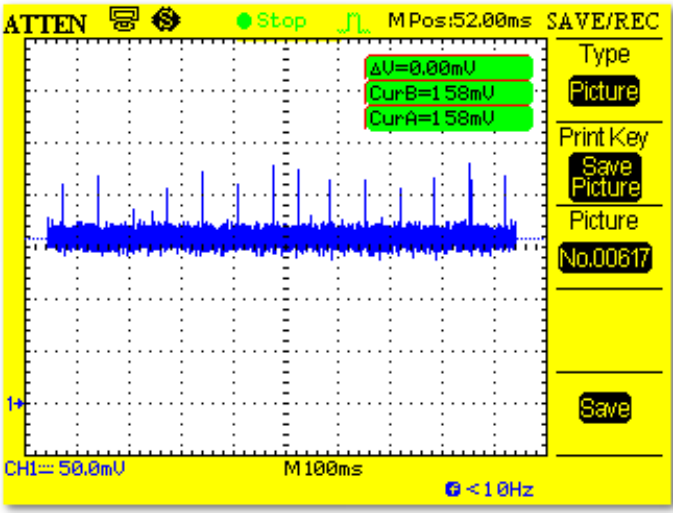


Fig. C.73. Medida general proceso de lectura sensor

Como se ve, la señal sale continua ya que el consumo de este sensor es prácticamente inapreciable.

El consumo del proceso total para la lectura del sensor de temperatura se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	50	0.005	0.25

Tabla C.78. Consumo del proceso total para la lectura del sensor de temperatura

C.2.6.9. Sensor de luminosidad

Este ejemplo muestra como leer el sensor LDR conectado a la placa de Smart Metering. Para hacer esto, el sensor tiene que encenderse previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

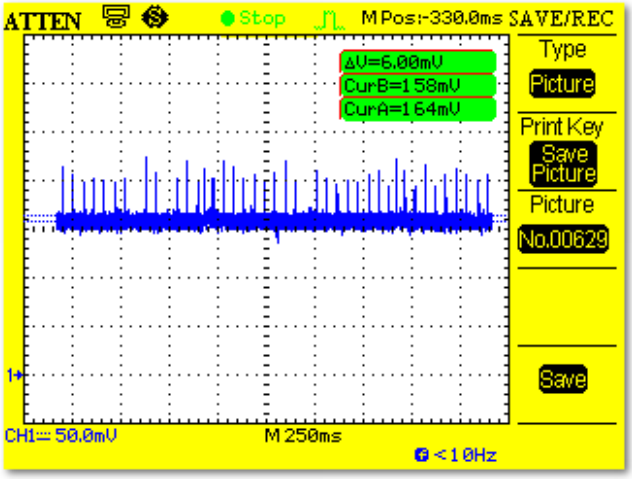


Fig. C.74.a. Medida general proceso

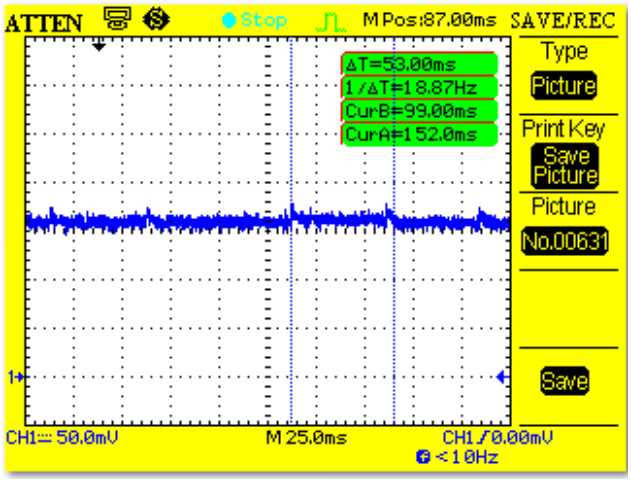


Fig. C.74.b. Anchura proceso de lectura

El consumo del proceso total para la lectura del sensor de luminosidad se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Proceso total	50	0.6	30

Tabla C.79. Consumo del proceso total para la lectura del sensor LDR

C.2.6.10. Sensor de lámina de desplazamiento alimentado a 3.3V

Este ejemplo muestra como leer el sensor de lámina de desplazamiento conectado al socket de 3.3V. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

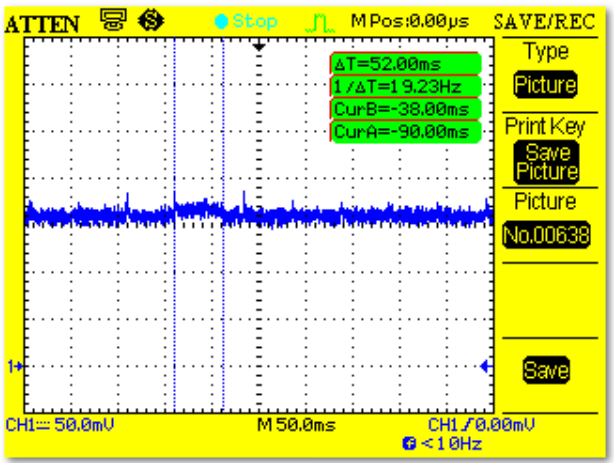


Fig. C.75.a. Anchura proceso de lectura

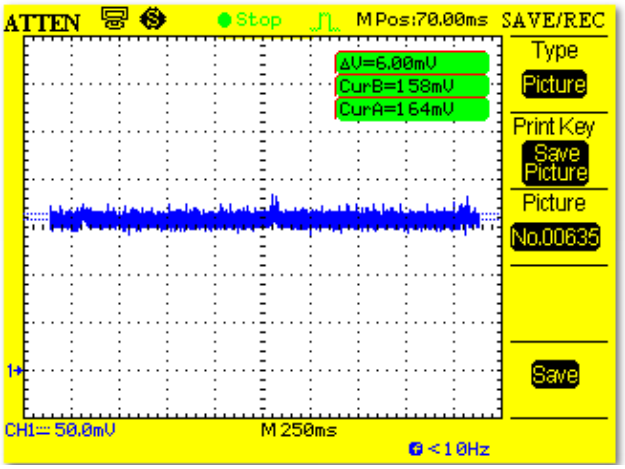


Fig. C.75.b. Medida general proceso

El consumo del proceso total para la lectura del sensor de lámina de desplazamiento se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	52	0.6	31.2

Tabla C.80. Consumo del proceso total para la lectura del sensor de lámina de desplazamiento

C.2.6.11. Sensor de lámina de desplazamiento alimentado a 5V

Este ejemplo muestra como leer el sensor de lámina de desplazamiento conectado al socket de 5V. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

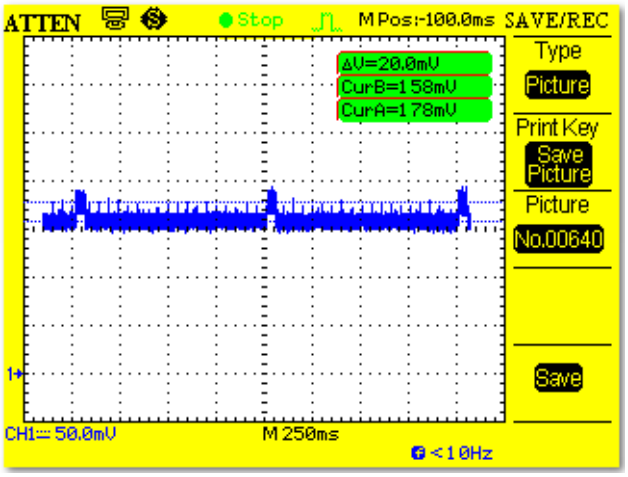


Fig. C.76.a. Medida general proceso

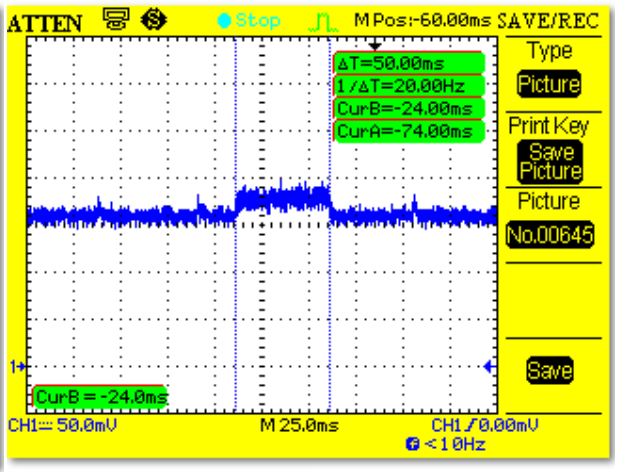


Fig. C.76.b. Anchura proceso de lectura

El consumo del proceso total para la lectura del sensor de lámina de desplazamiento se muestra en la siguiente tabla:

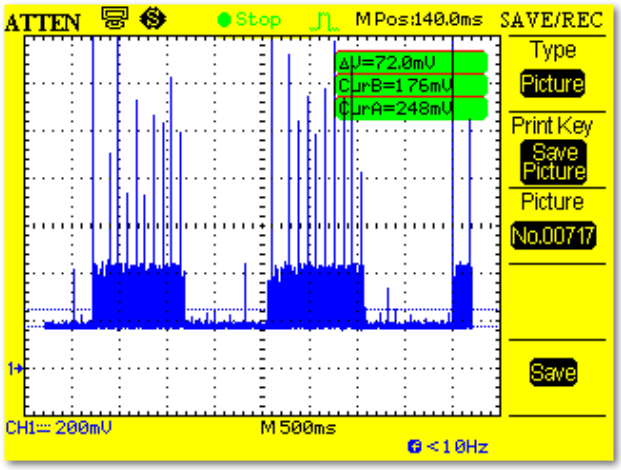
	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	50	2	100

Tabla C.81. Consumo del proceso total para la lectura del sensor de lámina de desplazamiento

C.2.6.12. Sensor Ultrasonido WRA1 alimentado a 5V

Este ejemplo muestra como leer el sensor de ultrasonido WRA1 conectado al socket de 5V de la placa. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:



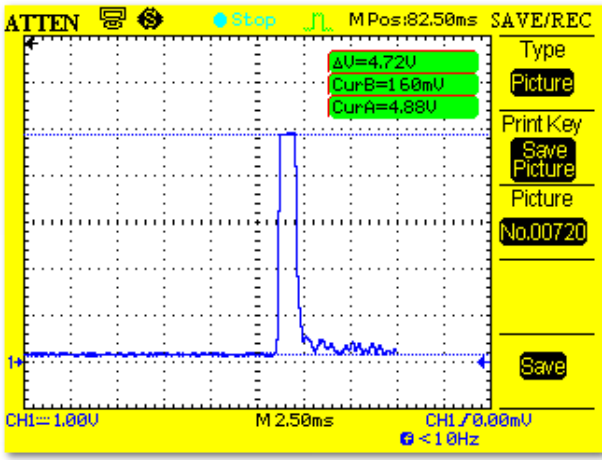
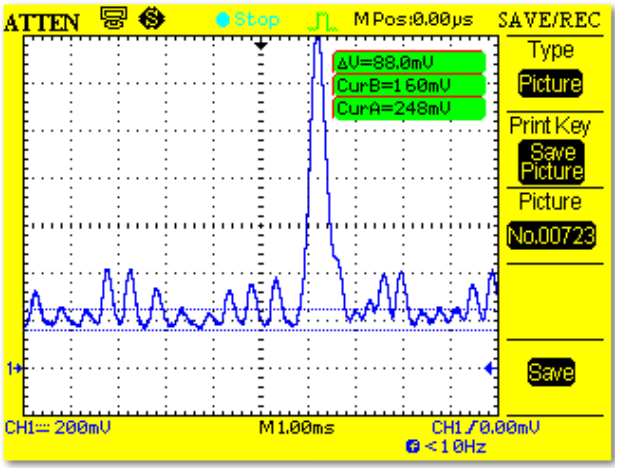


Fig. C.77.a. Medida general proceso cada segundo Fig. C.77.b. Pico de carga encendido sensor



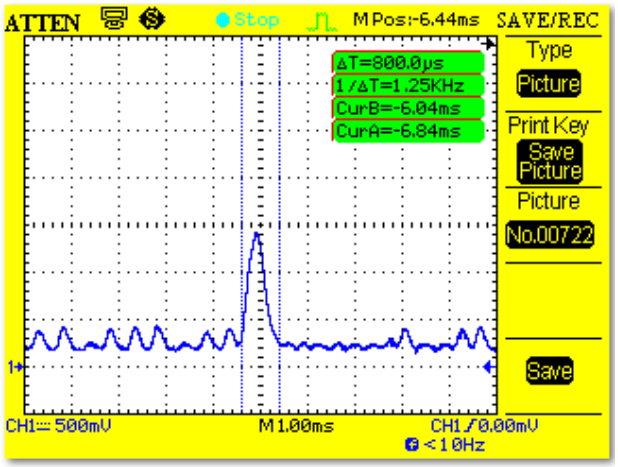


Fig. C.77.c. Pico durante el proceso de lectura

Fig. C.77.d. Anchura pico durante el proceso de lectura

El consumo del proceso total para la lectura del sensor WRA1 se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	1000	9.52	9522.04

Tabla C.82. Consumo del proceso total para la lectura del sensor WRA1

C.2.6.13. Sensor Ultrasonido EZO alimentado a 5V

Este ejemplo muestra como leer el sensor de ultrasonido EZO conectado al socket de 5V de la placa. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

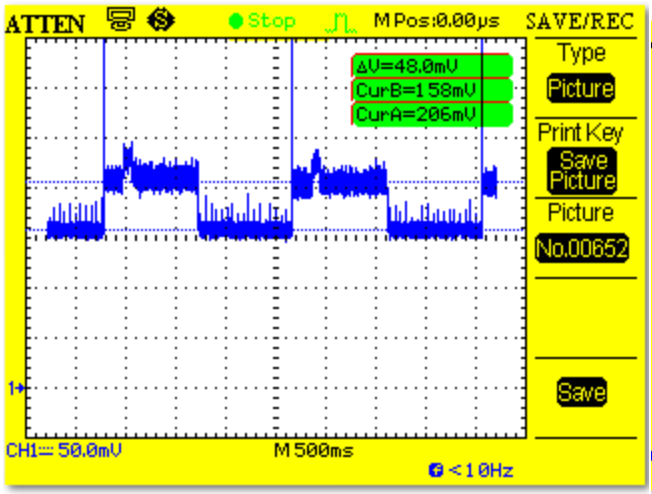


Fig. C.78.a. Medida general proceso cada segundo

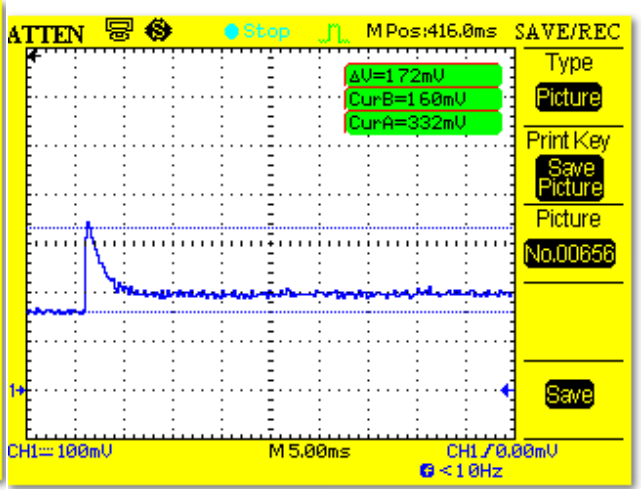


Fig. C.78.b. Pico de carga encendido sensor

El consumo del proceso total para la lectura del sensor EZO se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Proceso total	1000	4.815	4815.2

Tabla C.83. Consumo del proceso total para la lectura del sensor EZO

C.2.7. Placa de Prototipado

C.2.7.1. Proceso de encendido y apagado de la placa de prototipado

Este ejemplo muestra como encender y apagar la placa de prototipado cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

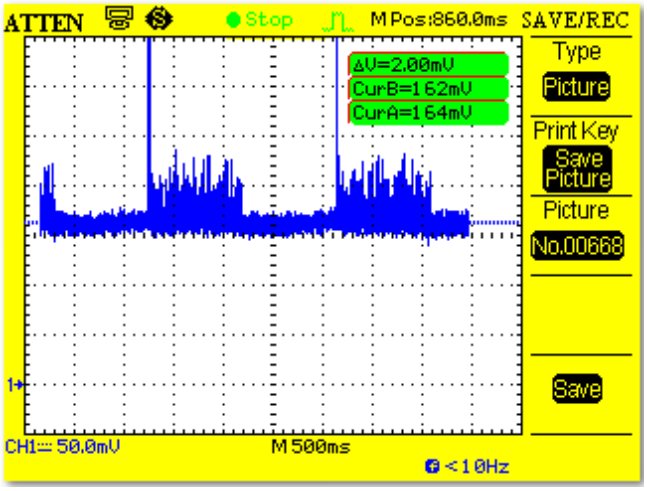


Fig. C.79.a. Medida general proceso cada segundo

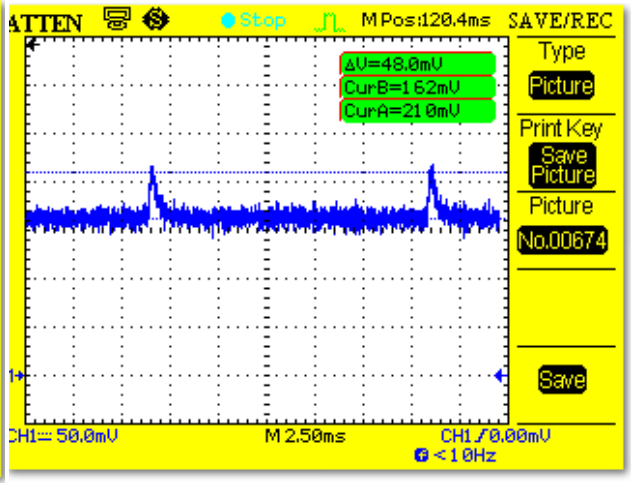


Fig. C.79.b. Picos durante el estado on

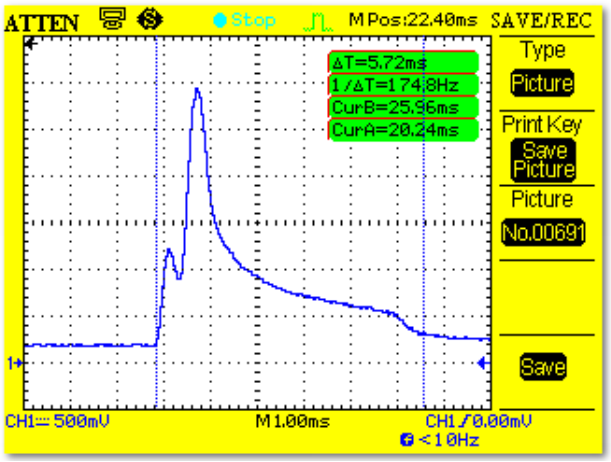


Fig. C.79.c. Achura pulso carga del proceso on

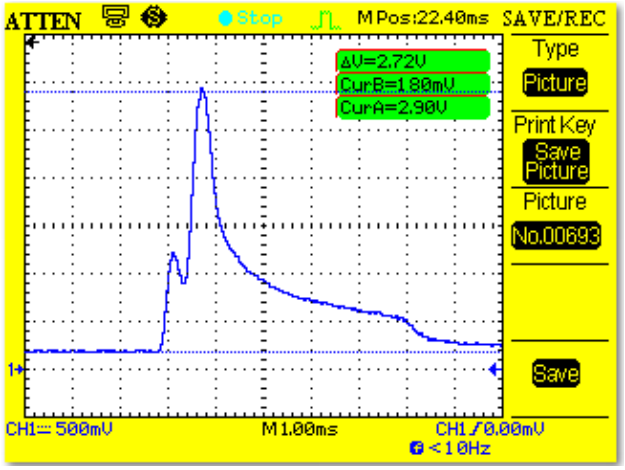


Fig. C.79.d. Altura pulso carga del proceso on

Los picos que aparecen en el estado On de la placa son debidos al conversor dc-dc. El gordo del principio es debido a la carga de condensadores, y luego aparecen picos, periódicos, con mayor frecuencia cuanto mayor sea la corriente de salida del conversor.

Como se ve en las gráficas, se produce un consumo extra de 600 uA en el estado off de la placa. Para el cálculo del consumo se ha tomado como referencia el estado on del Waspmote, es decir, los 15.6 mA.

El consumo del proceso y estado on y proceso off de la placa de prototipado se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga Energética total (mA·ms)**
Proceso on	5.72	82	469.05
Estado off	1000	0.82	822.6
Proceso off	0	0.82	0

Tabla C.84. Consumo del proceso y estado on y proceso off de la placa de prototipado

C.2.7.2. Lectura del ADC

Este ejemplo muestra como leer el ADC de la placa de prototipado.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

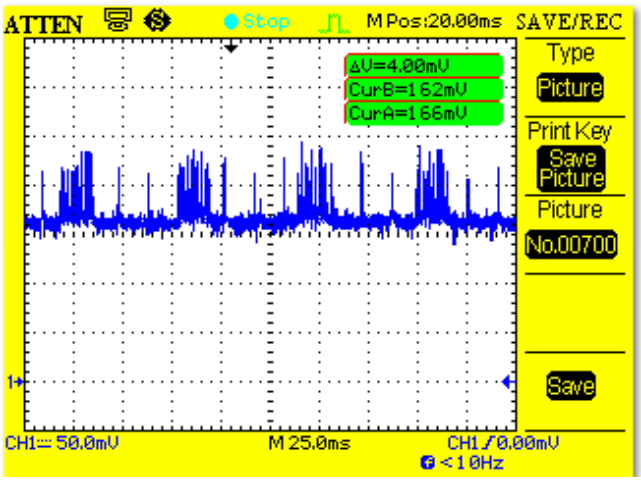


Fig. C.80.a. Medida general del proceso

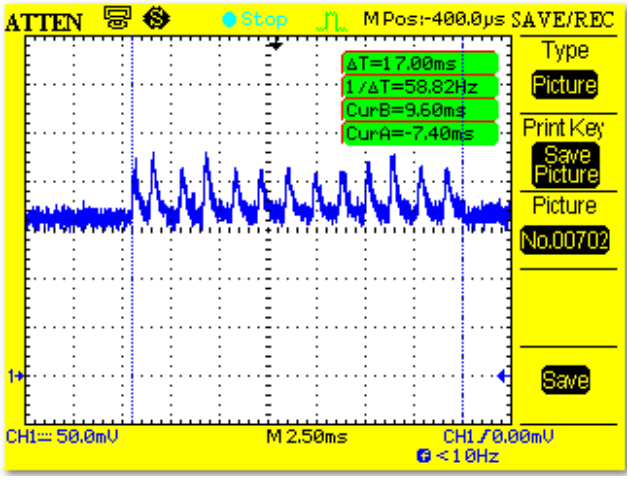


Fig. C.80.b. Tiempo de conversión del ADC

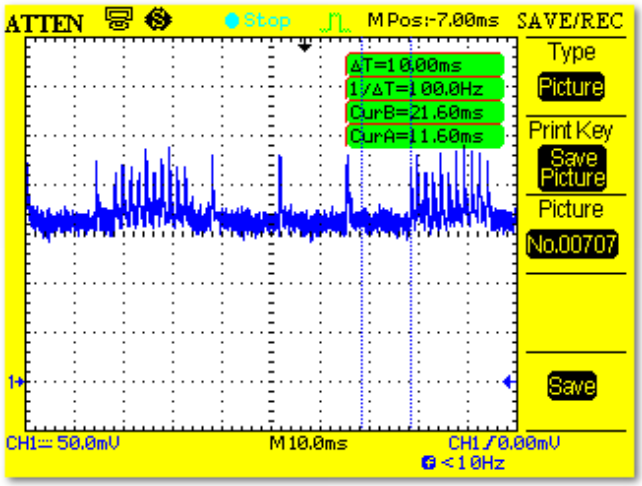


Fig. C.80.c. Tiempo necesario para estabilización

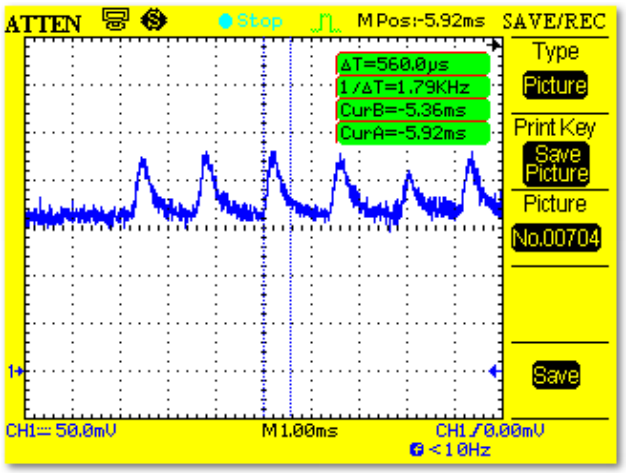


Fig. C.80.d. Archura pulsos durante la conversión

El consumo de la lectura del ADC se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga Energética total (mA·ms)**
Lectura ADC	27	1	27.18

Tabla C.85. Consumo lectura del ADC

C.2.8. Placa de Smart Cities

C.2.8.1.- Proceso de encendido y apagado de la placa de Smart Cities

Este ejemplo muestra como encender y apagar la placa de Smart Cities cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

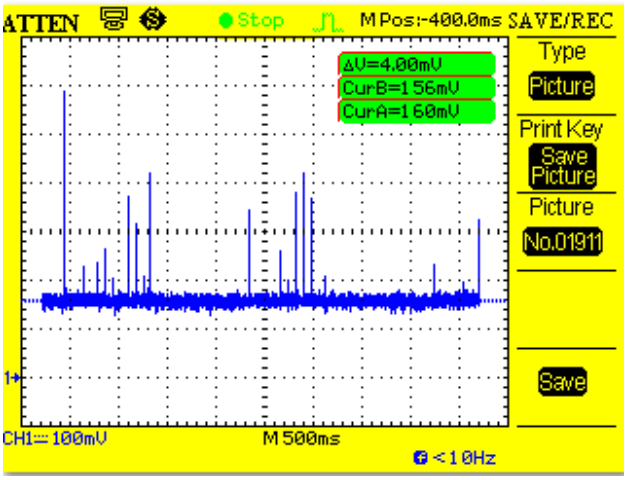


Fig. C.81.a. Medida general del proceso

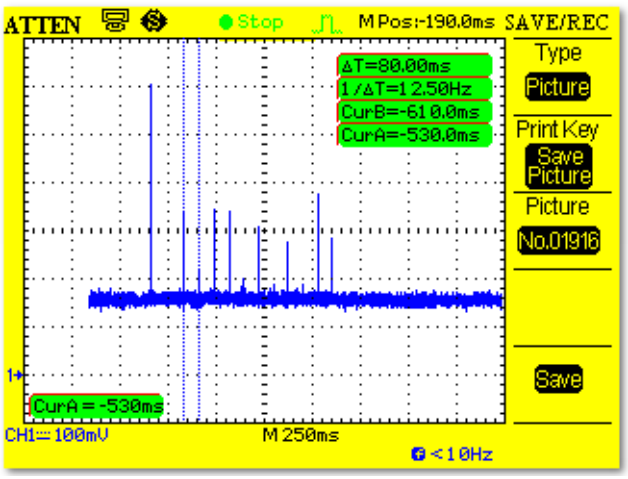


Fig. C.81.b. Anchura entre pulso y pulso

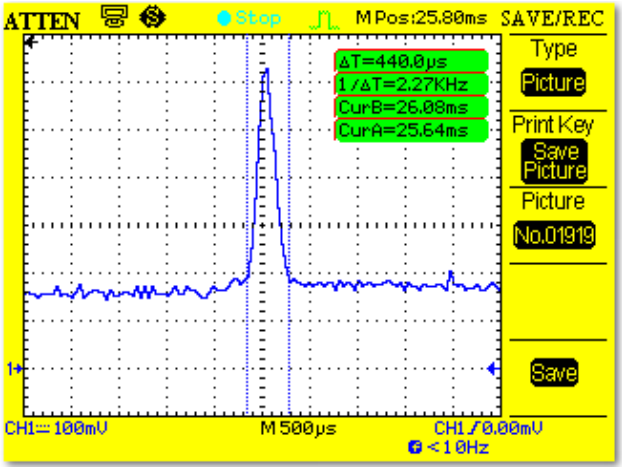


Fig. C.81.c. Anchura pico de carga proceso On

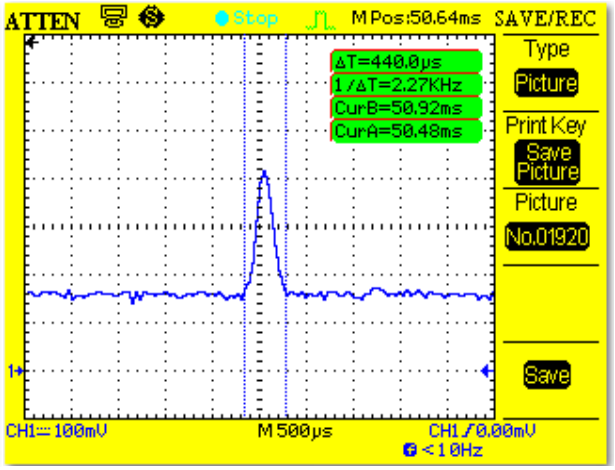


Fig. C.81.d. Anchura picos durante el estado On

Los picos que aparecen en el estado On de la placa son debidos al conversor dc-dc. El gordo del principio es debido a la carga de condensadores, y luego aparecen picos, periódicos, con mayor frecuencia cuanto mayor sea la corriente de salida del conversor.

El consumo del proceso y estado On y el proceso off de la placa de Smart Cities se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga Energética total (mA·ms)
On process	0.44	33.12	14.6
On state	1000	0.45	450.07
Off process	0	0.45	0

Tabla C.86. Consumo del proceso y estado on y proceso off de la placa de Smart Cities

C.2.8.2.- Sensor de temperatura

Este ejemplo muestra la lectura del sensor de temperatura conectado en la placa de Smart Cities cada segundo. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

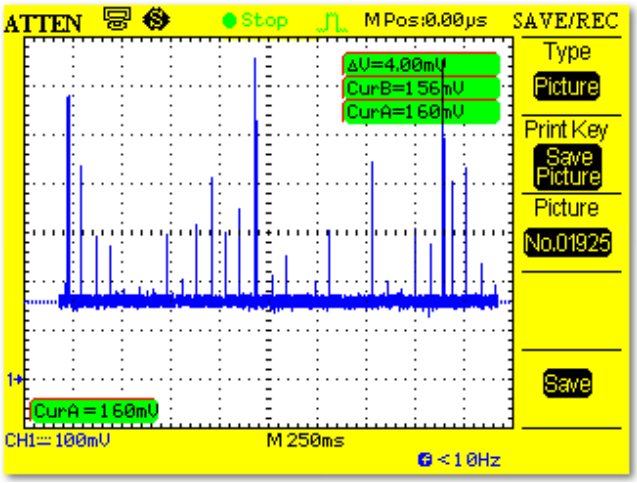


Fig. C.82.a. Medida general del proceso

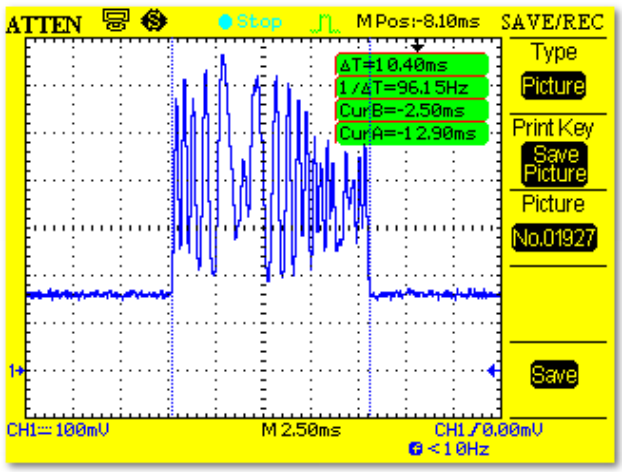


Fig. C.82.a. Anchura proceso lectura sensor

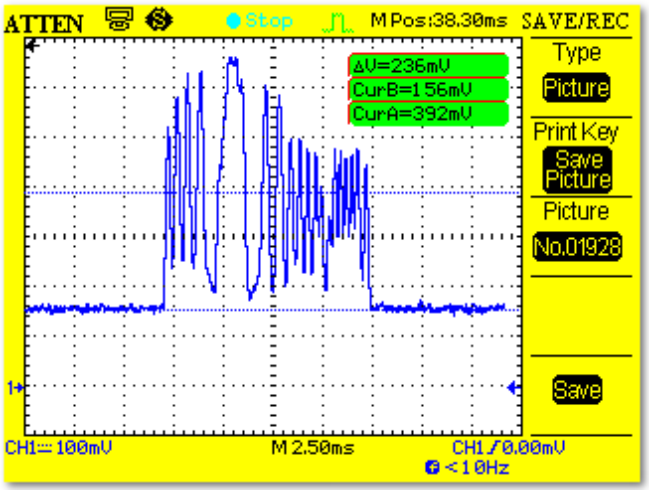


Fig. C.82.c. Altura proceso lectura sensor

El consumo del proceso total para la lectura del sensor de temperatura se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga Energética total (mA·ms)**
Proceso total	10	23.15	231.5

Tabla C.87. Consumo del proceso total para la lectura del sensor de temperatura

C.2.8.3. - Sensor de humedad

Este ejemplo muestra como leer el sensor de humedad conectado a la placa de Smart Cities cada 16 segundos. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

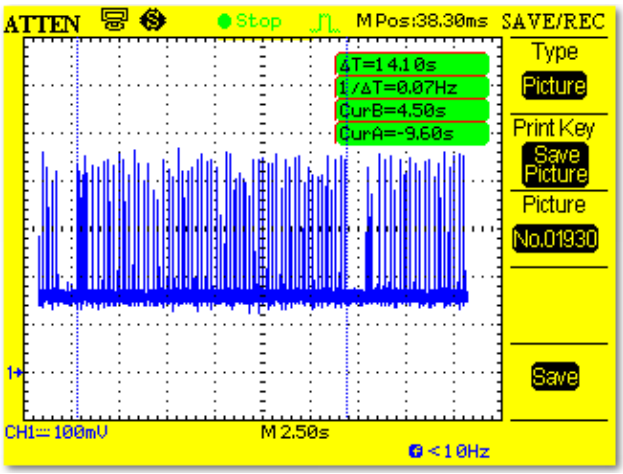


Fig. C.83.a. Medida general del proceso

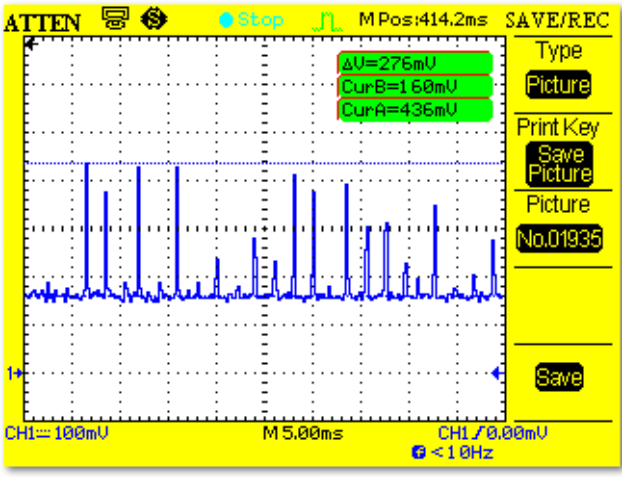


Fig. C.83.b. Altura pulsos durante el proceso

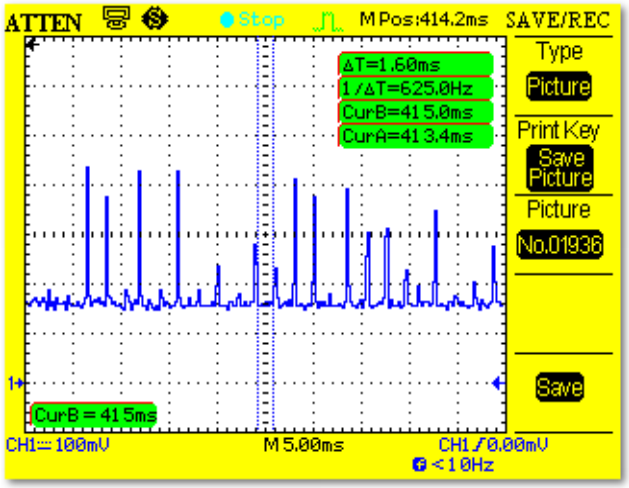


Fig. C.83.c. Anchura entre pulso y pulso

El consumo del proceso total para la lectura del sensor de humedad se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga Energética total (mA·ms)
Proceso total	15000	2.01	30282

Tabla C.88. Consumo del proceso total para la lectura del sensor de humedad

C.2.8.4. - Sensor LDR

Este ejemplo muestra como leer el sensor LDR conectado a la placa de Smart Cities cada segundo. Para hacer esto, el sensor se tiene que enceder previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

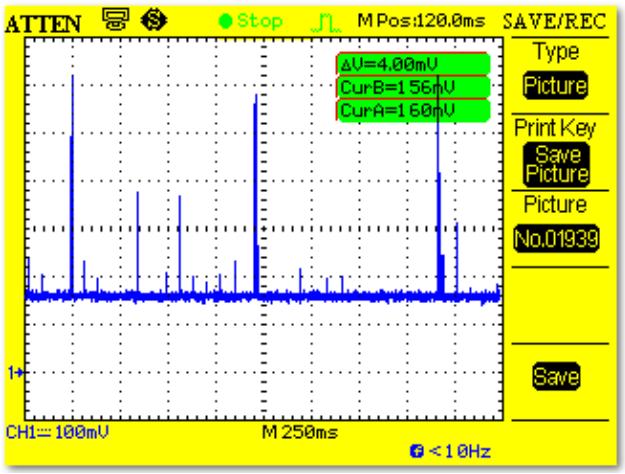


Fig. C.84.a. Medida general del proceso

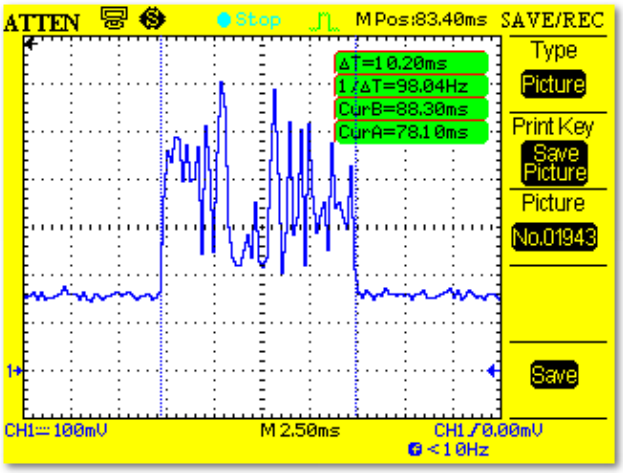


Fig. C.84.a. Anchura proceso lectura sensor

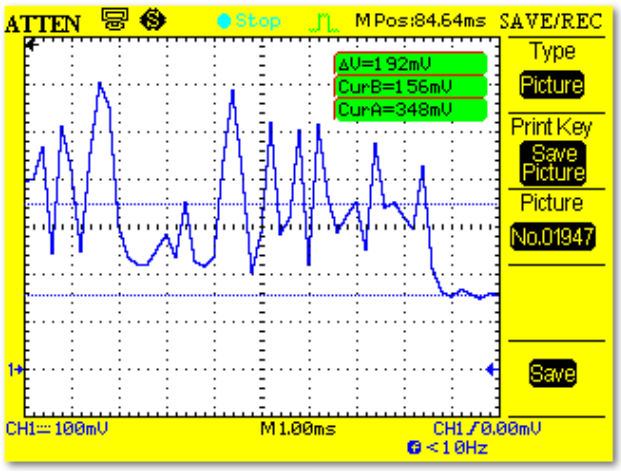


Fig. C.84.c. Altura proceso lectura sensor

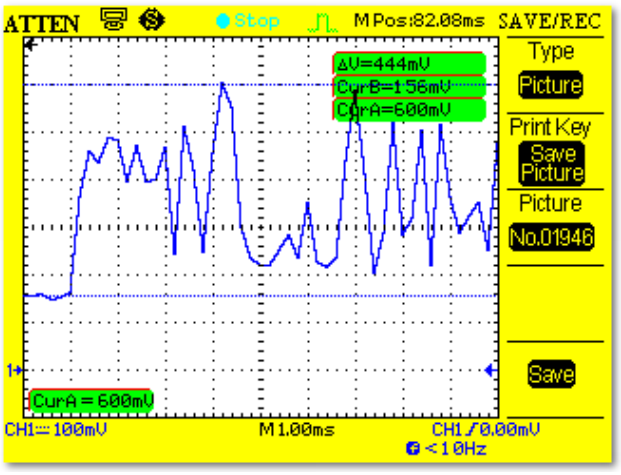


Fig. C.84.d. Altura máxima proceso lectura

El consumo del proceso total para la lectura del sensor LDR se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga Energética total (mA·ms)
Proceso total	10	19.95	199.5

Tabla C.89. Consumo del proceso total para la lectura del sensor LDR

C.2.8.5. - Sensor de partículas

Este ejemplo muestra como leer el sensor de partículas conectado a la placa de Smart Cities cada tres segundos. Para hacer esto, el sensor tiene que encenderse previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

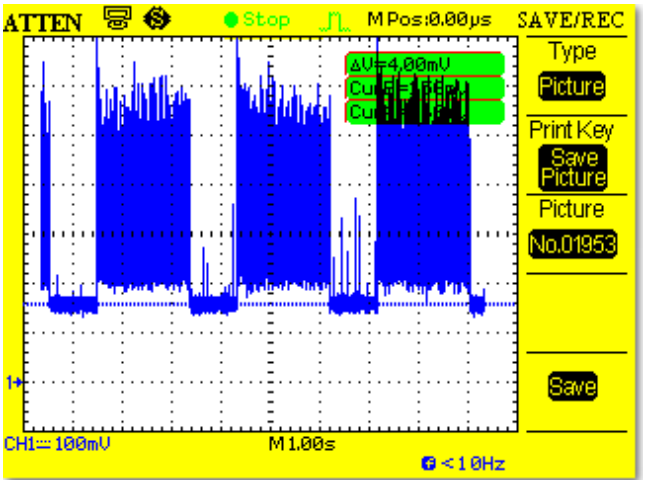


Fig. C.85.a. Medida general del proceso

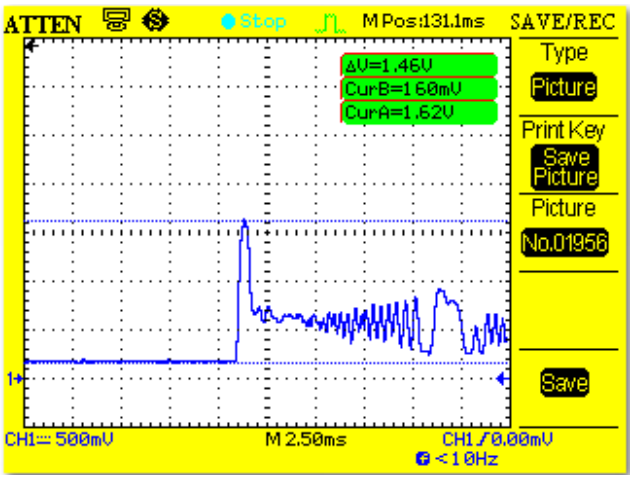
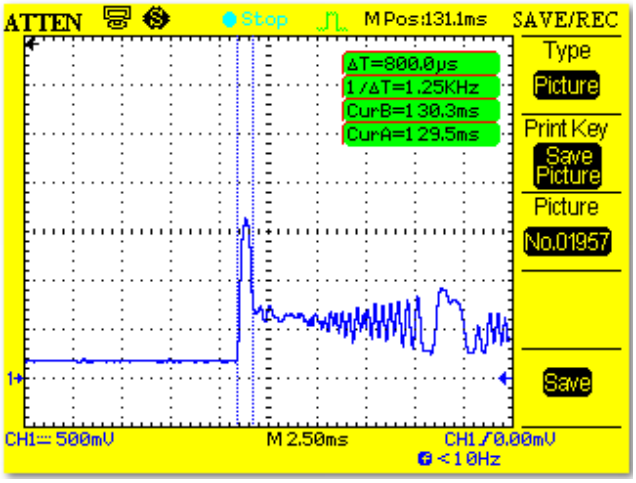
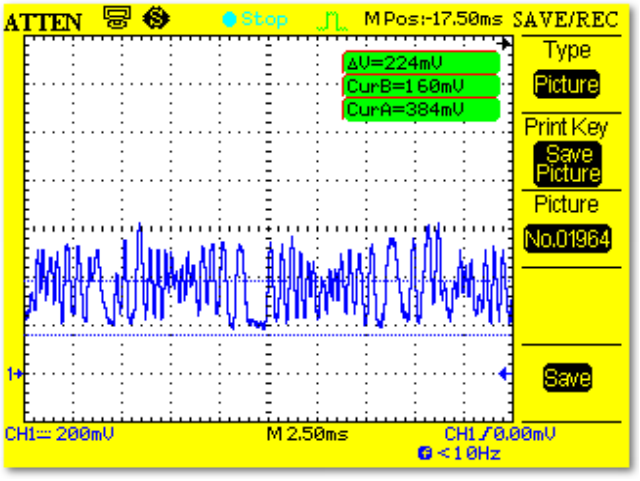


Fig. C.85.b. Altura pulso de carga del sensor



C.85.c. Anchura pulso de carga del sensor



C.85.d. Altura del proceso de lectura del sensor

El consumo del proceso total para la lectura del sensor de partículas se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga Energética total (mA·ms)
Proceso total	2000	22.45	44898.88

Tabla C.90. Consumo del proceso total para la lectura del sensor de partículas

C.2.8.6. - Sensor de detección de grietas

Este ejemplo muestra como leer el sensor de detección de grietas, conectado a la placa de Smart Cities cada segundo. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

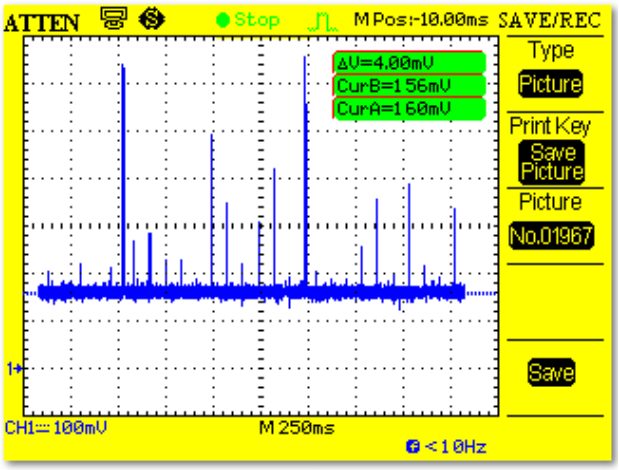


Fig. C.86.a. Medida general del proceso

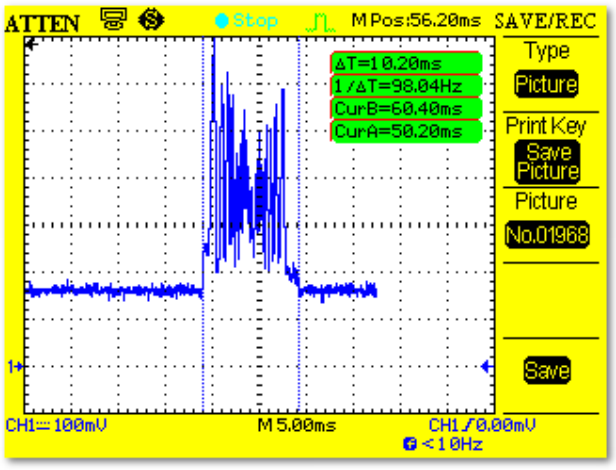


Fig. C.86.b. Anchura proceso lectura sensor

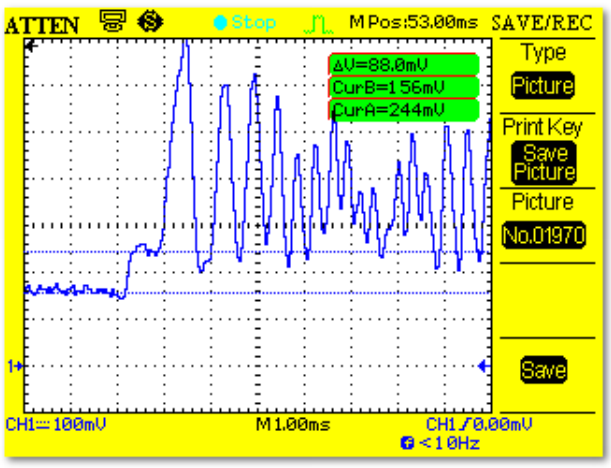


Fig. C.86.c. Inicio proceso

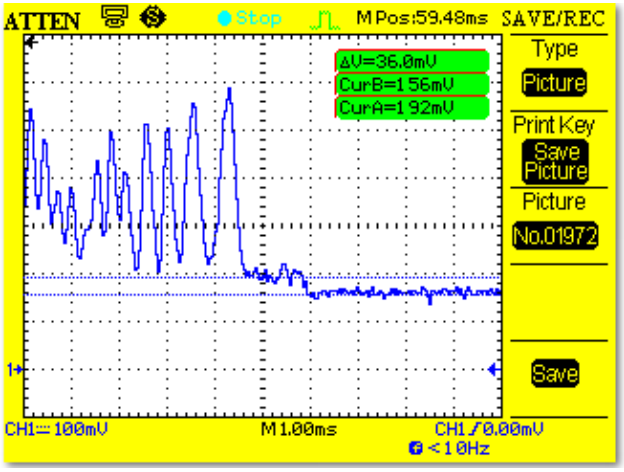


Fig. C.86.d. Final proceso

El consumo del proceso total para la lectura del sensor de detección de grietas se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga Energética total (mA·ms)
Proceso total	10	18.48	184.892

Tabla C.91. Consumo del proceso total para la lectura del sensor de detección de grietas

C.2.8.7. - Sensor de ruido

Este ejemplo muestra como leer el sensor de ruido conectado a la placa de Smart Cities cada tres segundos. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

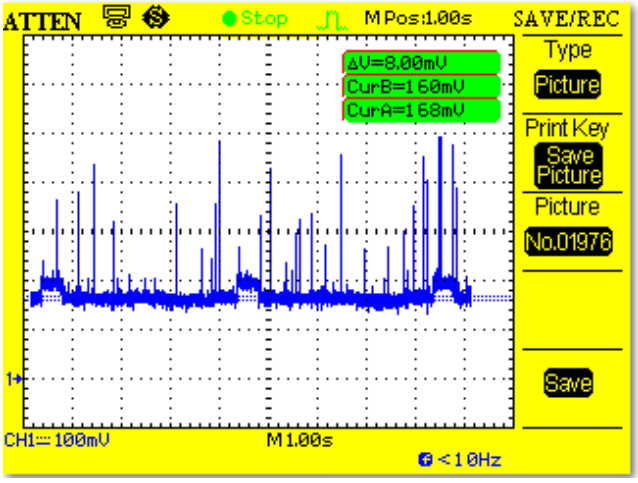


Fig. C.87.a. Medida general del proceso

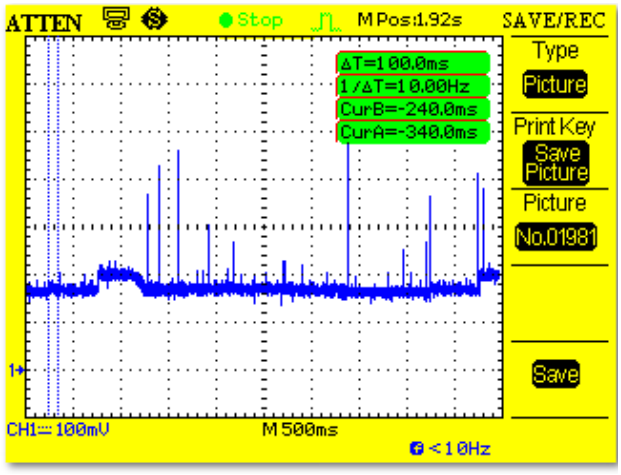


Fig. C.87.b. Tiempo proceso on del sensor

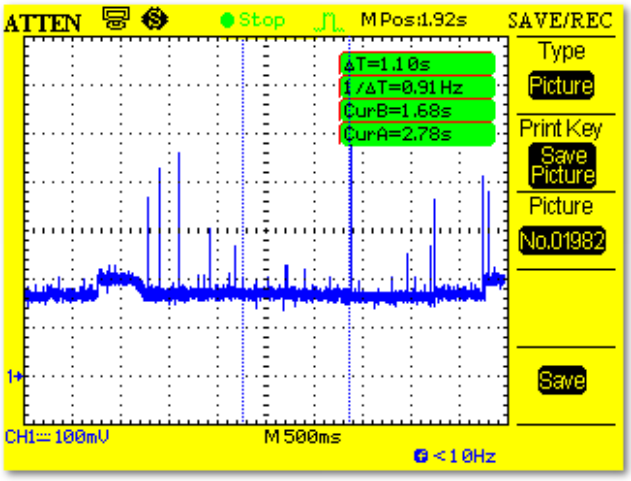


Fig. C.87.c. Tiempo de ejecución de la lectura

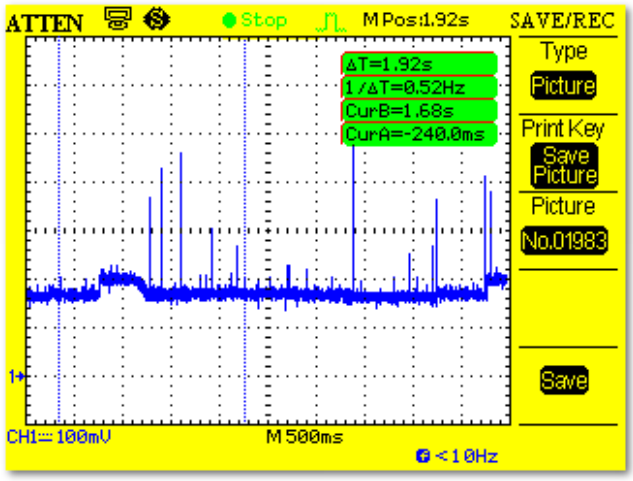


Fig. C.87.d. Tiempo de estabilización del sensor

El consumo del proceso total para la lectura del sensor de detección de ruido se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga Energética total (mA·ms)**
Proceso total	3200	1.2	3848

Tabla C.92. Consumo del proceso total para la lectura del sensor de ruido

C.2.8.8. – Sensor de ultrasonido WRA1 a 3.3V

Este ejemplo muestra como leer el sensor de ultrasonido WRA1 a 3.3V conectado en la placa de Smart Cities cada tres segundos. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

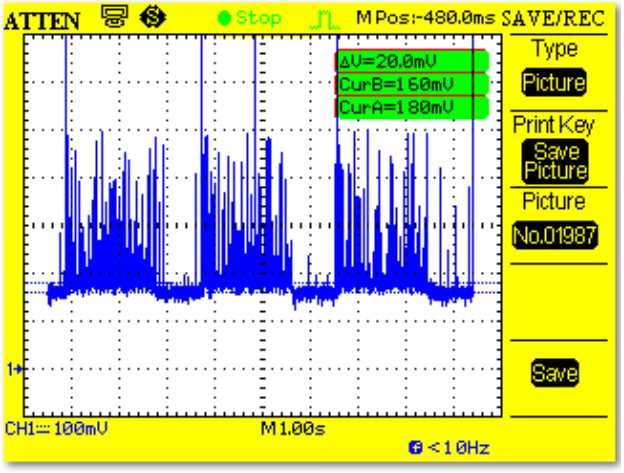


Fig. C.88.a. Medida general del proceso

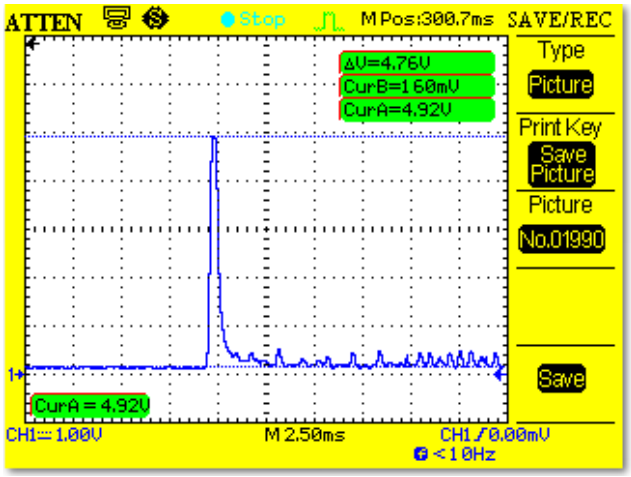


Fig. C.88.b. Altura del pulso de carga

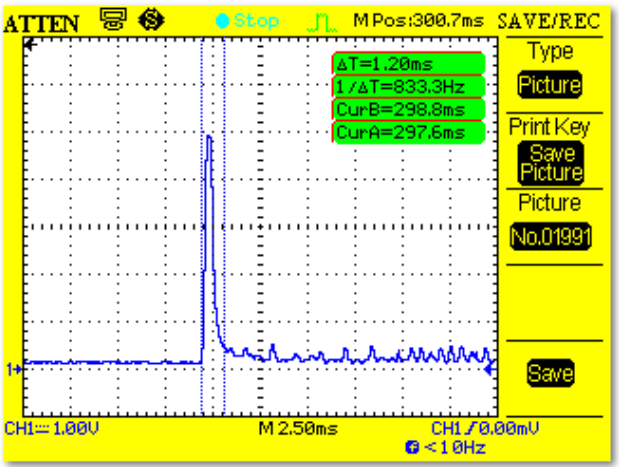


Fig. C.88.c. Anchura del pulso de carga

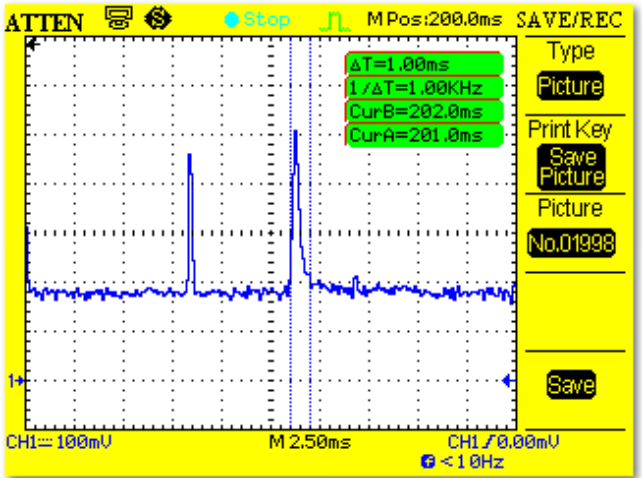


Fig. C.88.d. Anchura pulsos durante el Proceso

El consumo del proceso total para la lectura del sensor de ultrasonido WRA1 se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga Energética total (mA·ms)**
Proceso total	2000	2.99	5992.64

Tabla C.93. Consumo del proceso total para la lectura del sensor de ultrasonido

C.2.8.9. – Sensor de ultrasonido EZO a 3.3V

Este ejemplo muestra como leer el sensor de ultrasonido EZO a 3.3V conectado en la placa de Smart Cities cada tres segundos. Para hacer esto, el sensor se tiene que encender previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

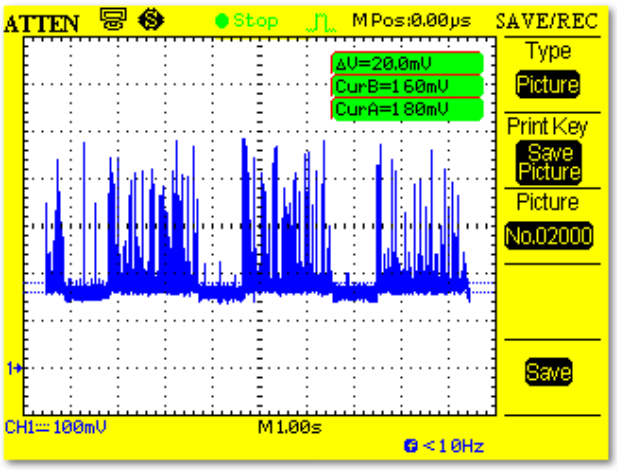


Fig. C.89.a. Medida general del proceso

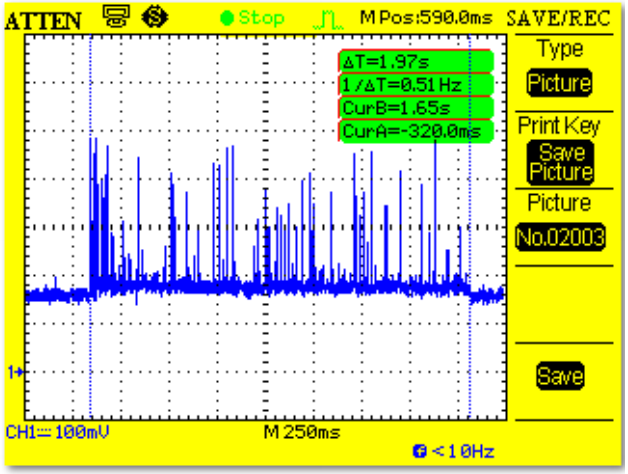


Fig. C.89.b. Anchura total proceso

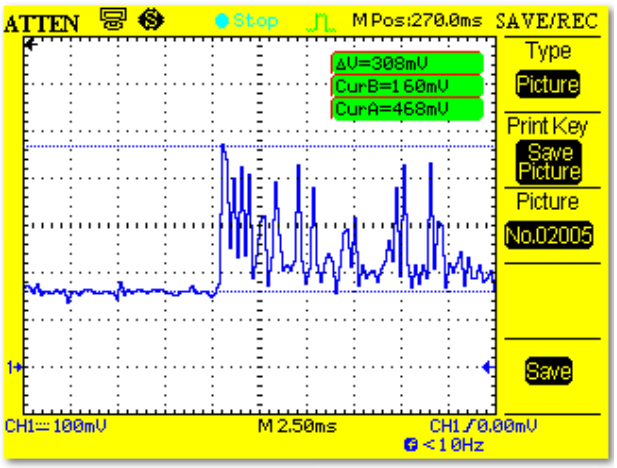


Fig. C.89.c. Inicio proceso lectura sensor

El consumo del proceso total para la lectura del sensor de ultrasonido EZO se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga Energética total (mA·ms)**
Proceso total	2000	2.4	4808.96

Tabla C.94. Consumo del proceso total para la lectura del sensor de ultrasonido

C.2.9. Placa de Radiación

C.2.9.1.- Proceso de encendido y apagado de la placa de Radiación

Este ejemplo muestra como encender y apagar la placa de Radiación.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

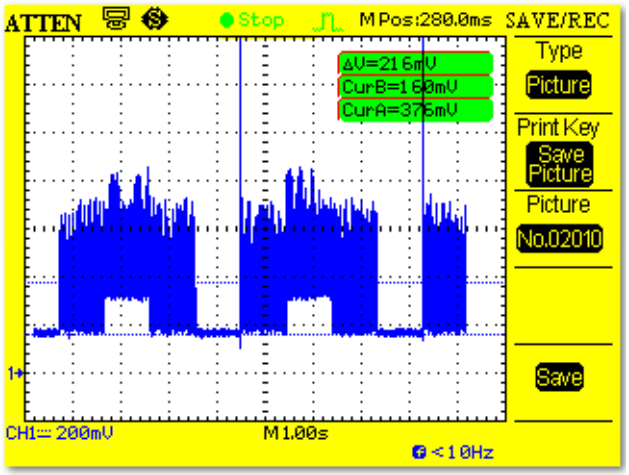


Fig. C.90.a.Medida general cada segundo

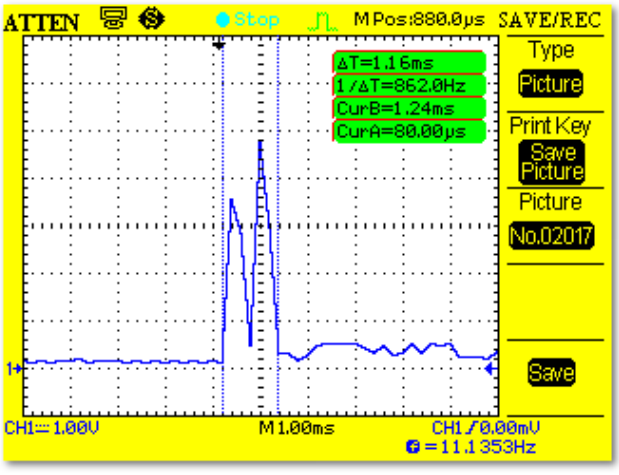


Fig. C.90.b. Duración del pico del on de la placa

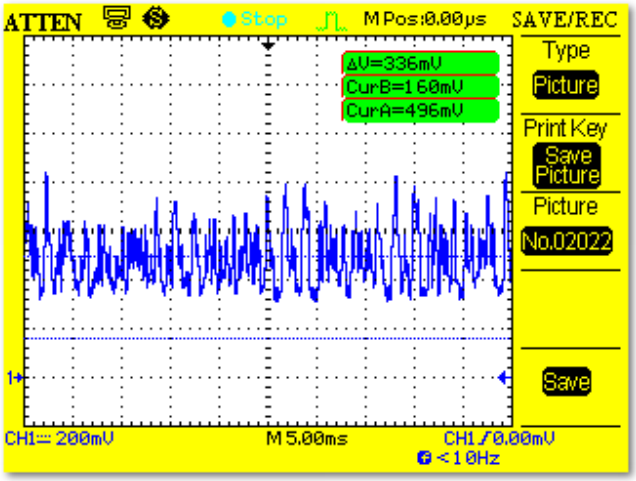


Fig. C.90.c. Consumo parte alta del proceso on

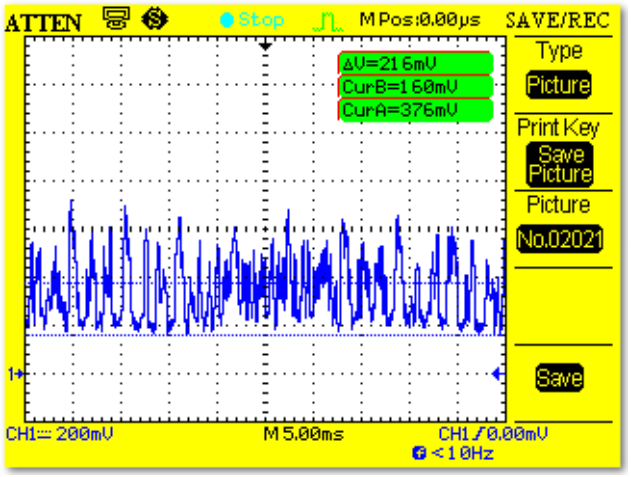


Fig. C.90.d. Consumo del estado on

Como se ve en las gráficas, el proceso a On de la placa le cuesta un tiempo de ejecución de 2000 ms mientras que el proceso a Off se hace instantáneo.

El consumo del proceso y estado on y proceso off de la placa de radiación se muestra en la siguiente tabla:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	2000	28.14	56291.25
Estado On	1000	22	22000
Proceso Off	0	22	0

Tabla C.95. Consumo en los diferentes procesos del On/off placa de radiación

C.2.9.2.- Lectura de la radiación en cpm

Este ejemplo muestra como leer la radiación en cpm (counts per minute) durante cinco segundos a través del tubo Geiger-Muller con un valor de radiación entre 150 y 400 cmp encendiéndose dos leds verdes.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

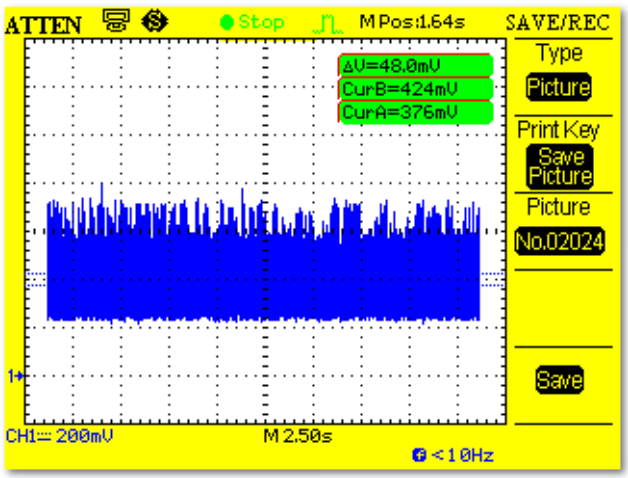


Fig. C.91.a. Medida general del proceso

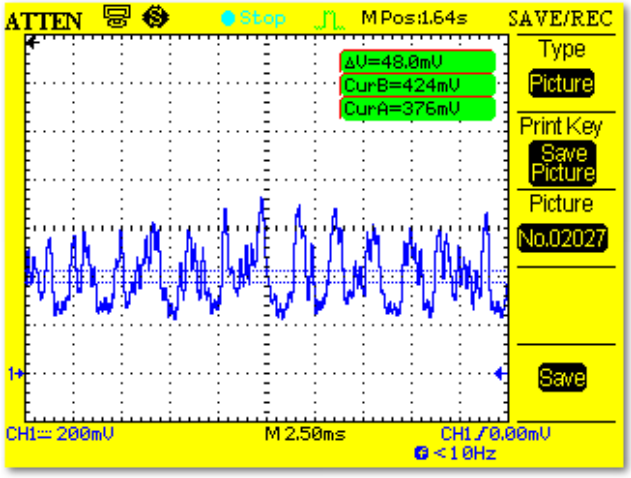


Fig. C.91.b. Consumo proceso de lectura de la radiación

Cabe comentar que el valor de la corriente depende de la radiación detectada. Si la radiación aumenta, el consumo energético también aumentará.

El consumo del proceso total para la lectura de radiación se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga Energética total (mA·ms)**
Proceso total	5000	4.8	24000

Tabla C.96. Consumo del proceso total para la lectura de radiación

C.2.10. Placa de Smart Water

C.2.10.1.- Proceso de encendido y apagado de la placa de Smart Water

Este ejemplo muestra como encender y apagar la placa de Smart Water cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

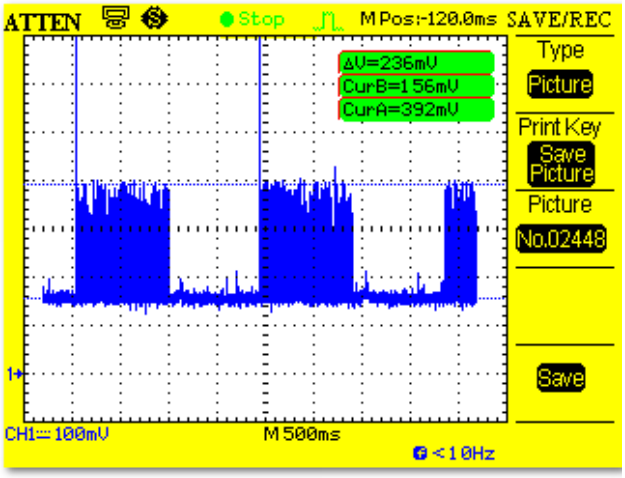


Fig. C.92.a. Medida general del proceso

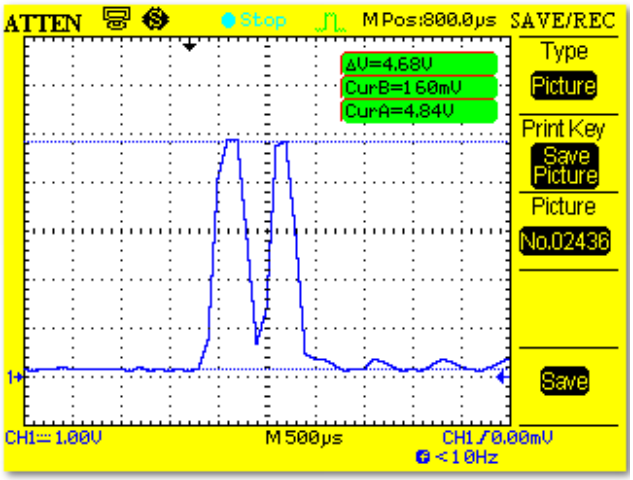


Fig. C.92.b. Altura pico de carga proceso on

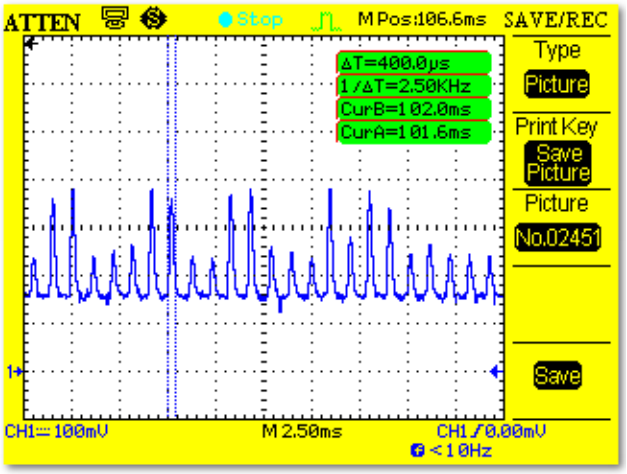


Fig. C.92.c. Anchura pulsos durante el estado on

Los picos que aparecen en el estado On de la placa son debidos al conversor dc-dc. El gordo del principio es debido a la carga de condensadores, y luego aparecen picos, periódicos, con mayor frecuencia cuanto mayor sea la corriente de salida del conversor.

El consumo del proceso y estado on y proceso off de la placa de Smart Water se muestra en la siguiente tabla:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	1	300.61	300.61
Estado On	1000	4.03	4032
Proceso Off	0	4.03	0

Tabla C.97. Consumo en los diferentes procesos del On/off placa de Smart Water

C.2.10.2.- Sensor Pt1000

Este ejemplo muestra como leer el sensor Pt1000 de la placa de Smart Water.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

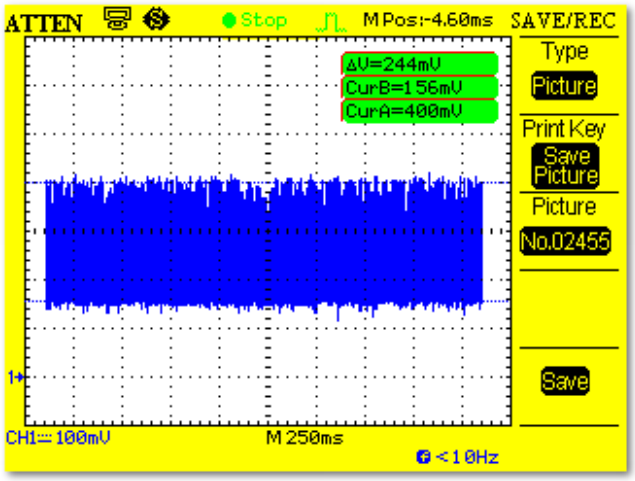


Fig. C.93.a.Medida general del proceso

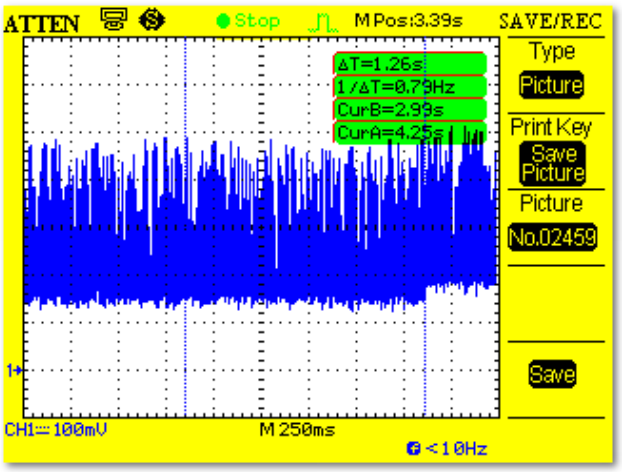


Fig. C.93.b. Tiempo de ejecución de la lectura del sensor

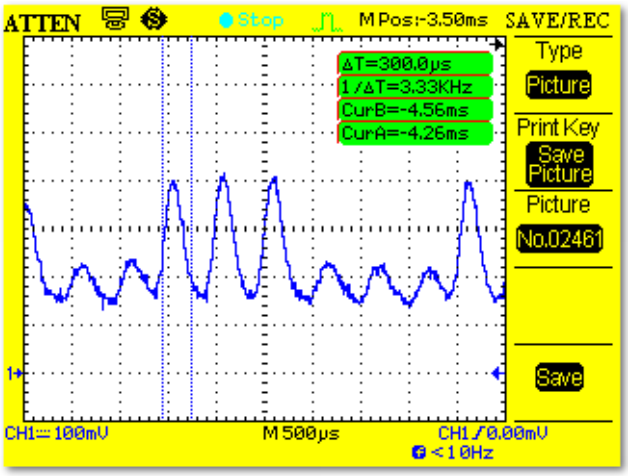


Fig. C.93.c. Anchura pulsos

Como se ve en la figura (b), para saber con exactitud cuanto es el tiempo de ejecución de la lectura del sensor PT1000, se ha añadido una serie de funciones al código que establece un consumo extra a lo medido y así distinguirlo con facilidad.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Lectura Pt1000	1260	3.68	4636.8

Tabla C.98. Consumo de la lectura del sensor

C.2.10.3.- Sensor de Conductividad

Este ejemplo muestra como leer el sensor de conductividad de la placa de Smart Water.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

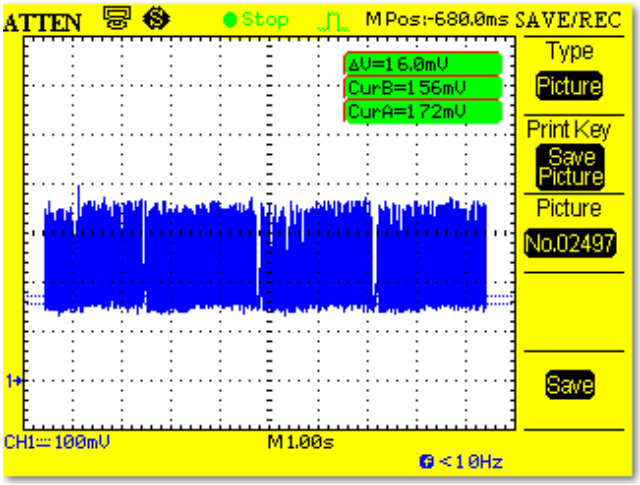


Fig. C.94.a. Medida general del proceso

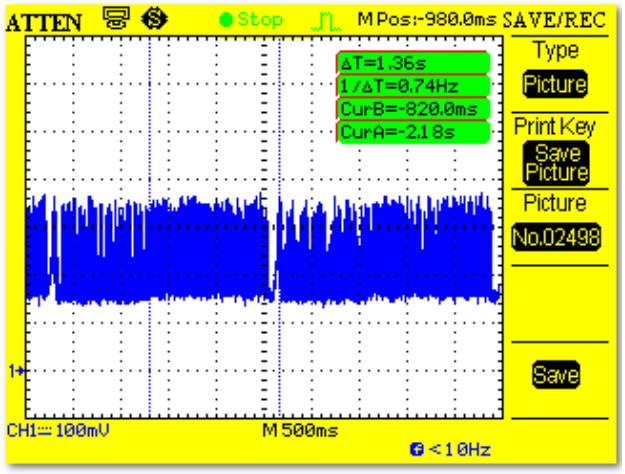


Fig. C.94.b. Tiempo de ejecución de la lectura del sensor

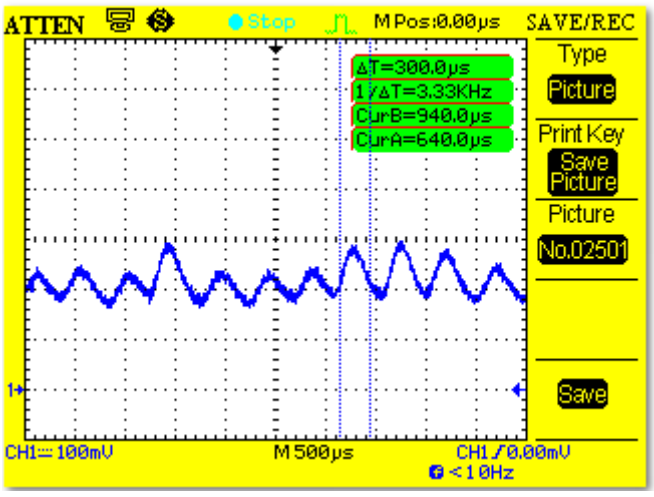


Fig. C.94.c. Anchura pulsos

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Lectura sensor conductividad	1360	2.37	3223.2

Tabla C.99. Consumo de la lectura del sensor

C.2.10.4. - Sensor de Oxigeno disuelto

Este ejemplo muestra como leer el sensor de oxigeno disuelto de la placa de Smart Water.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

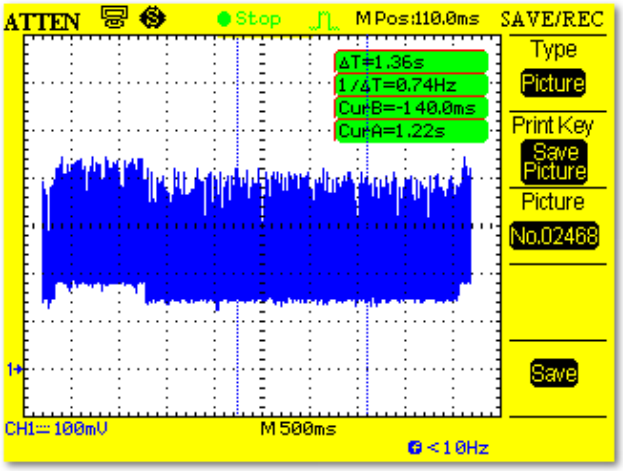


Fig. C.95.a. Tiempo ejecución lectura sensor

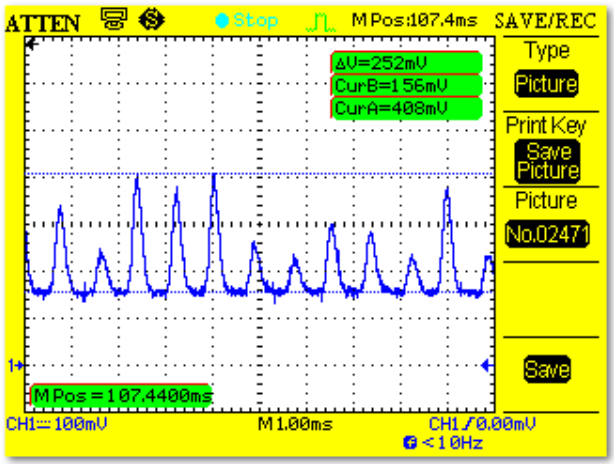


Fig. C.95.b. Picos de corriente durante el proceso

Como se ve en la figura (a), para saber con exactitud cuanto es el tiempo de ejecución de la lectura del sensor, se ha añadido una serie de funciones al código que establece un consumo extra a lo medido y así distinguirlo con facilidad.

El consumo de la lectura del sensor de oxigeno disuleto se muestra en la siguiente tabla:

	Tiempo total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Lectura del oxígeno disuelto	1360	0.825	1122

Tabla C.100. Consumo de la lectura del sensor

C.2.10.5. - Sensor pH

Este ejemplo muestra como leer el sensor pH extrayendo el valor desde los valores de calibración y compensación de temperatura de la placa de Smart Water.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

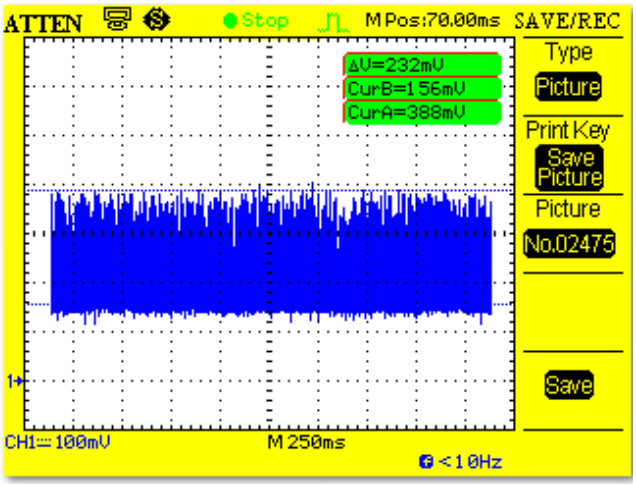


Fig. C.96.a. Medida general del proceso

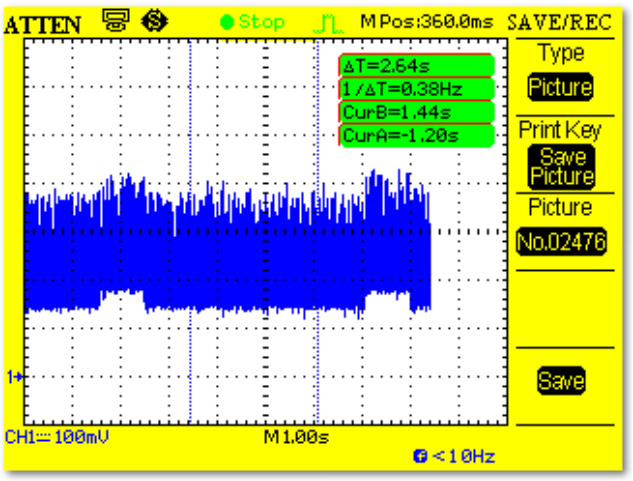


Fig. C.96.b. Tiempo ejecución lectura sensor

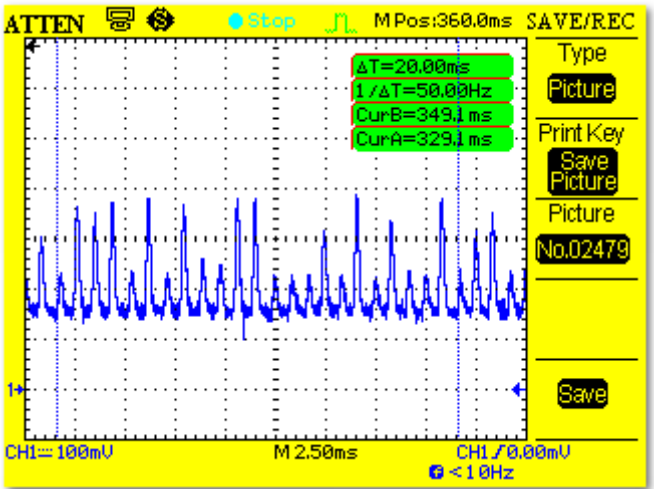


Fig. C.96.c. Picos de corriente durante el proceso

Como se ve en la figura (b), para saber con exactitud cuanto es el tiempo de ejecución de la lectura del sensor, se ha añadido una serie de funciones al código que establece un consumo extra a lo medido y así distinguirlo con facilidad.

El consumo de la lectura del sensor pH se muestra en la siguiente tabla:

	Tiempo total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Lectura pH	2640	0.92	2428.8

Tabla C.101. Consumo de la lectura del sensor

C.2.10.6. – Sensor potencial oxidación-reducción

Este ejemplo muestra como leer el sensor ORP extrayendo el valor desde el valor de calibración y compensación de temperatura de la placa de Smart Water.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

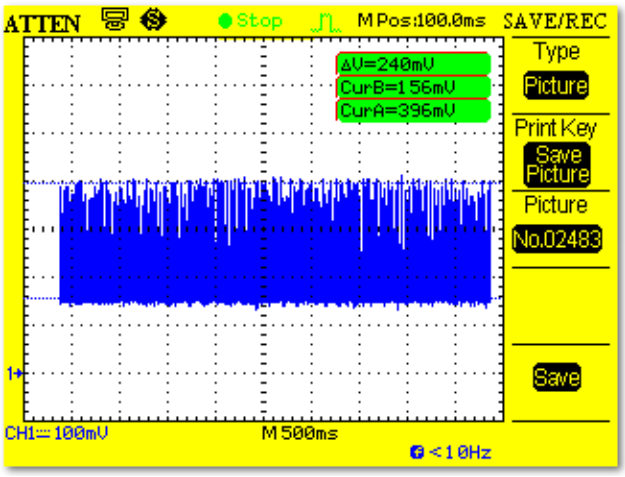


Fig. C.97.a. Medida general del proceso

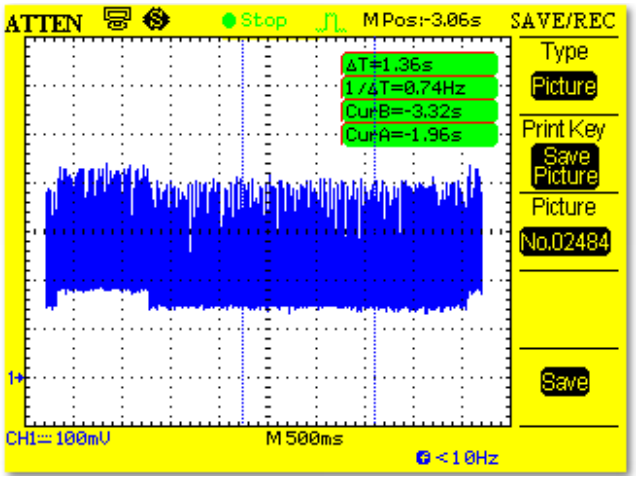


Fig. C.97.b. Tiempo ejecución lectura sensor

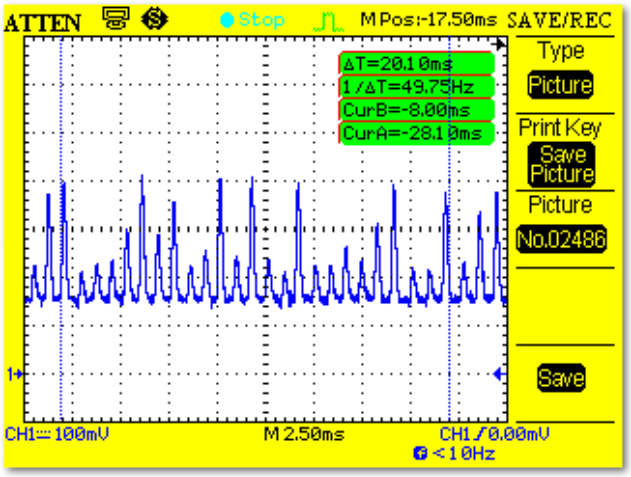


Fig. C.97.c. Picos de corriente durante el proceso

Como se ve en la figura (b), para saber con exactitud cuanto es el tiempo de ejecución de la

lectura del sensor, se ha añadido una serie de funciones al código que establece un consumo extra a lo medido y así distinguirlo con facilidad.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Tiempo total (ms)	Corriente total (mA)	Carga energética total (mA·ms)
Lectura sensor ORP	1360	0.932	1267.52

Tabla C.102. Consumo de la lectura del sensor

C.2.10.7. – Sensor de Iones Disueltos

Este ejemplo muestra como leer el sensor DI de la placa de Smart Water.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

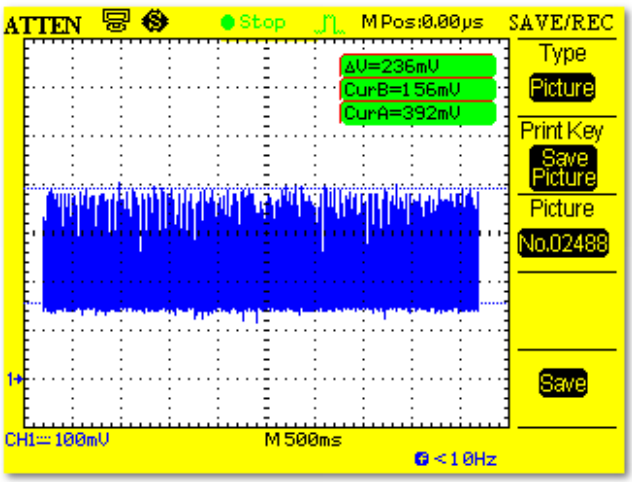


Fig. C.98.a. Medida general del proceso

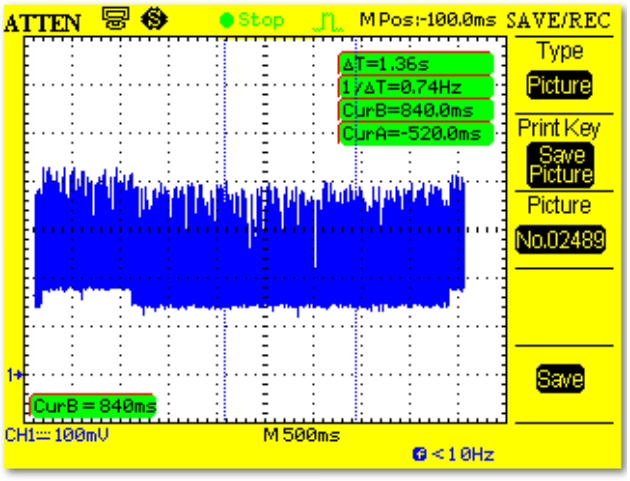


Fig. C.98.b. Tiempo de ejecución lectura sensor

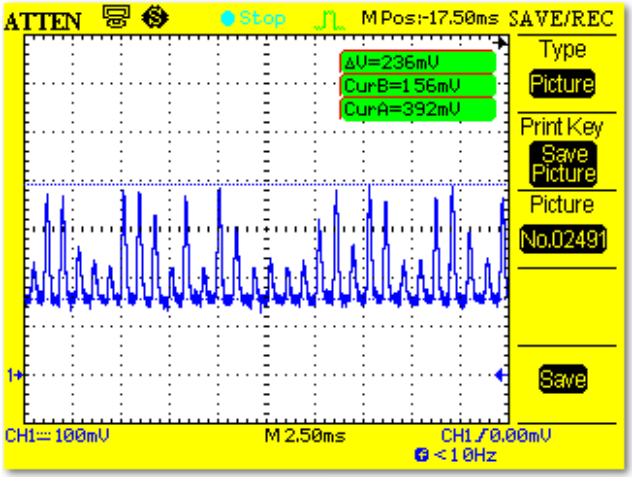


Fig. C.98.c. Picos de corriente durante el proceso

Como se ve en la figura (b), para saber con exactitud cuanto es el tiempo de ejecución de la lectura del sensor, se ha añadido una serie de funciones al código que establece un consumo extra a lo medido y así distinguirlo con facilidad.

El consumo de la lectura del sensor se muestra en la siguiente tabla:

	Tiempo total (ms)	**Corriente total (mA)**	**Carga energética total (mA·ms)**
Lectura sensor DI	1360	0.97	1319.2

Tabla C.103. Consumo de la lectura del sensor

C.2.11. Placa de Video Cámara

Para hacer las acciones necesarias en esta placa, es necesario conectar el módulo 3G también a la placa de Waspmote.

C.2.11.1.- Proceso de encendido y apagado de la placa de Videocámara

Este ejemplo muestra como encender y apagar la placa de Video Cámara cada segundo. Para usar este ejemplo, es necesario, previamente, encender el módulo 3G.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

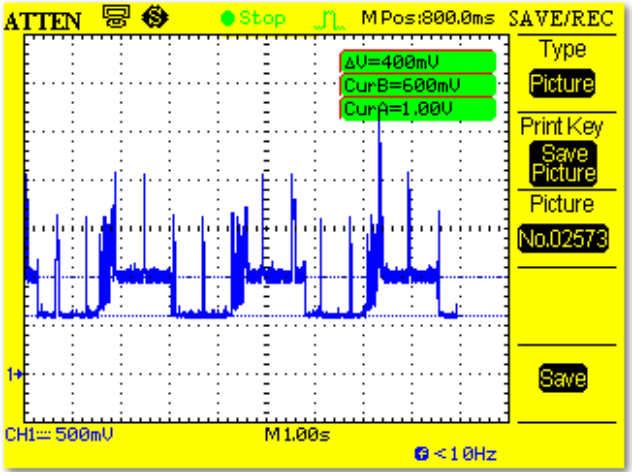


Fig. C.99.a. Medida general del proceso

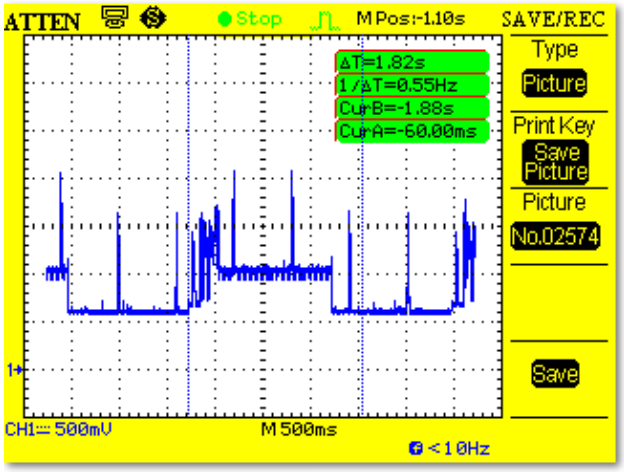


Fig. C.99.b. Tiempo ejecución total procesos

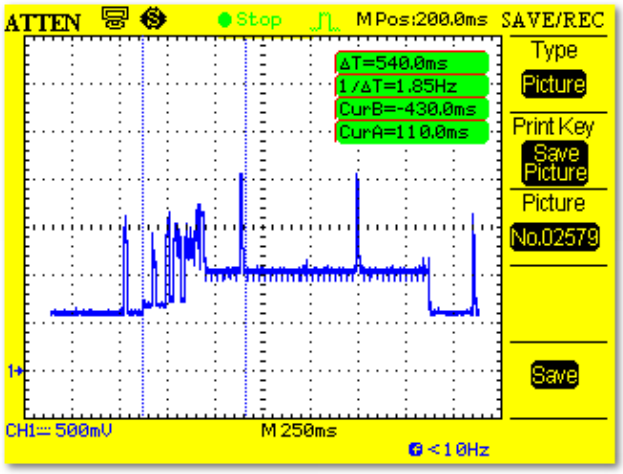


Fig. C.99.c. Tiempo ejecución proceso on

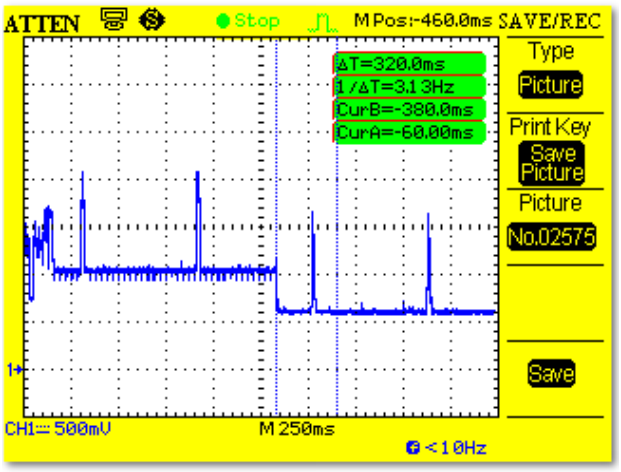


Fig. C.99.d. Tiempo ejecución proceso off

El consumo del proceso y estado on y proceso off de la placa de Video Cámara se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetica total (mA·ms)**
Proceso on	540	53.83	29072
Estado on	1000	44.256	44256
Proceso off	320	2	640

Tabla C.104. Consumo del proceso y estado on y proceso off de la placa de Video Cámara

C.2.11.2.- Proceso de encendido y apagado de los Leds IR

Este ejemplo muestra como encender y apagar los Leds IR de la placa de Video Cámara. Para usar este ejemplo, es necesario encender el módulo 3G y la placa previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

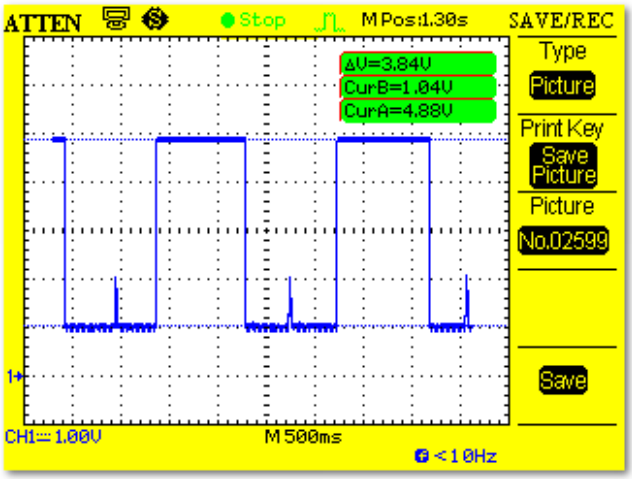


Fig. C.100.a. Medida genera del proceso

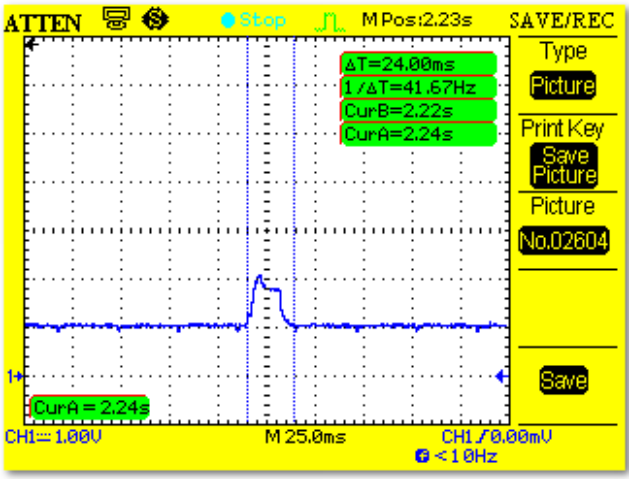


Fig. C.100.b. Anchura pulso

El consumo del proceso y estado on y proceso off de los leds IR se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energetica total (mA·ms)
Proceso on	0	384	0
Estado on	1000	384	384000
Proceso off	0	384	0

Tabla C.105. Consumo del proceso y estado on y proceso off de los leds IR

C.2.11.3.- Configuración de la cámara y grabar un video

Este ejemplo muestra como configurar la cámara y como grabar un video. Para usar este ejemplo, es necesario encender el módulo 3G y poner la tarjeta micro-SD en él previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

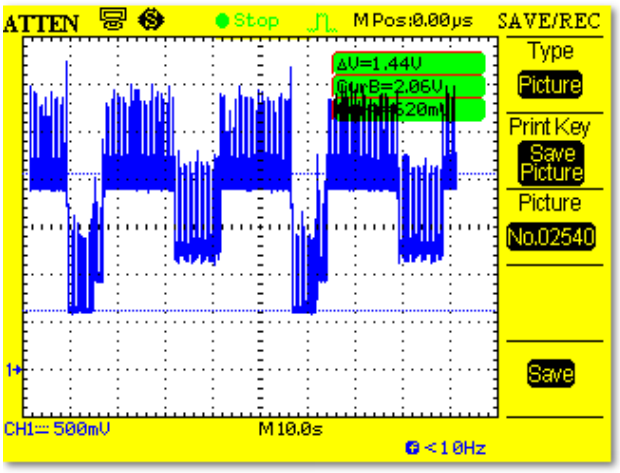


Fig. C.101.a. Medida general del proceso

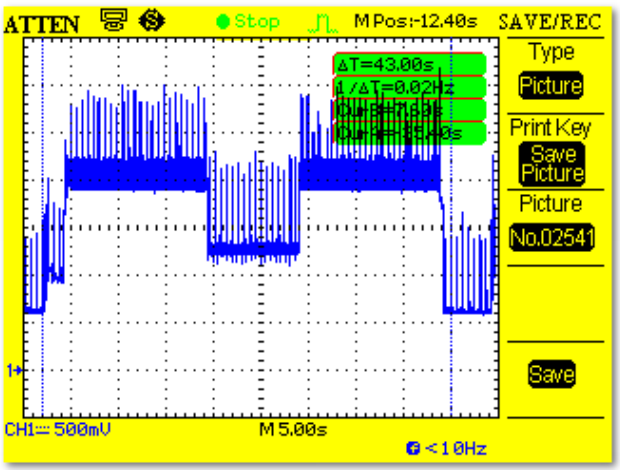


Fig. C.101.b. Tiempo ejecución total del proceso

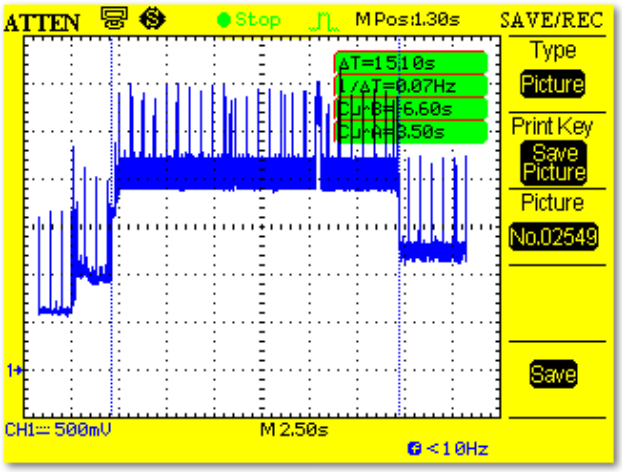


Fig. C.101.c. Tiempo de grabación del video

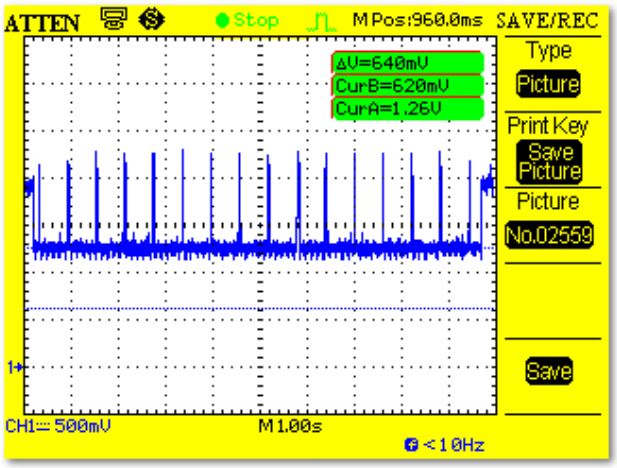


Fig. C.101.d. Tiempo pausa del video

El consumo de los diferentes procesos para grabar un video se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetica total (mA·ms)**
Camara on	540	53.83	29072
Configuración camara	1260	45.58	57432
Grabar video	41780	120.75	5045096
Camara off	320	2	640

Tabla C.106. Consumo en los diferentes procesos por los que pasa para grabar un video

El consumo del proceso total se muestra en la siguiente tabla:

	Duración total (ms)	**Corriente total (mA)**	**Carga energetica total (mA·ms)**
Proceso total	43900	116.9	5132240

Tabla C.107. Consumo del proceso total para grabar un vídeo

C.2.11.4.- Interrupción de PIR haciendo una foto y subiéndola al servidor FTP

Este ejemplo muestra cómo usar la interrupción de PIR haciendo una foto y subiéndola al servidor FTP cuando una interrupción ocurre. Para usar este ejemplo, es necesario encender el modulo 3G, poner la tarjeta micro-SD en él y poner correctamente el sevidor FTP, el puerto al que está dirigido, el nombre de usuario y la contraseña de acuerdo al servidor FTP creado previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

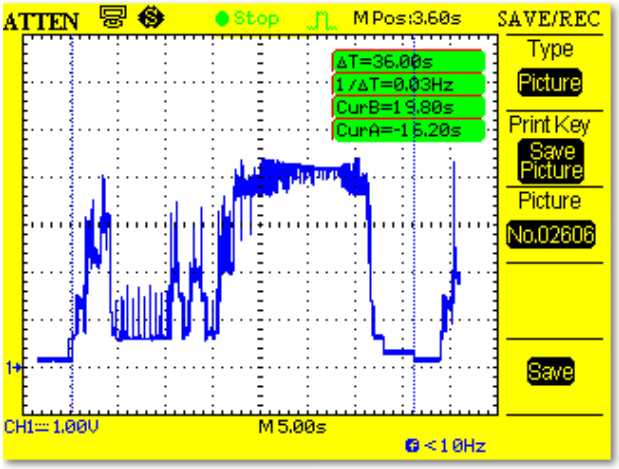


Fig. C.102.a. Tiempo ejecución proceso total

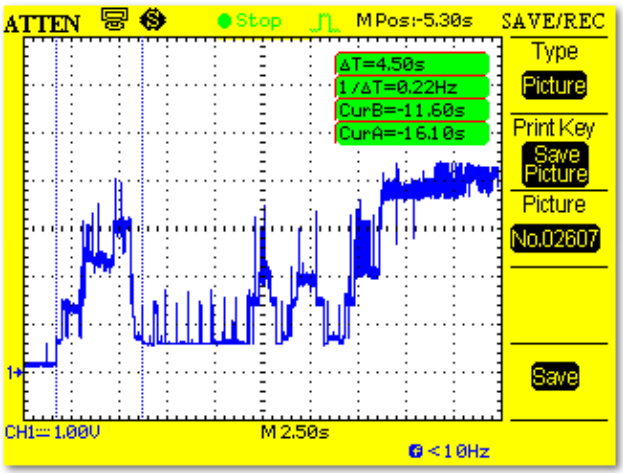


Fig. C.102.b. Proceso de habilitar la interrupción

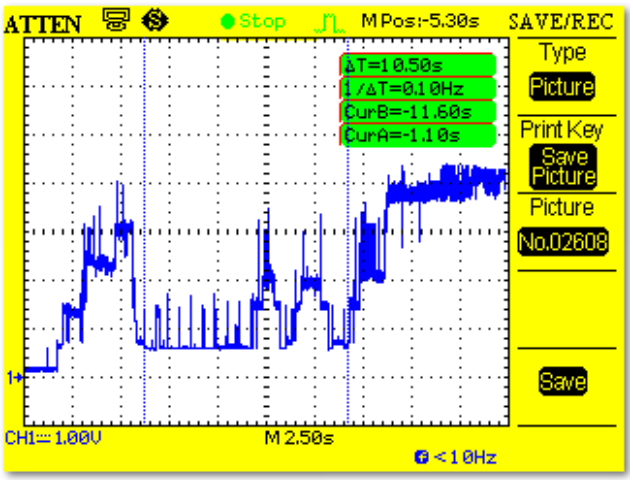


Fig. C.102.c. Conexión a la red y configuración del servidor FTP

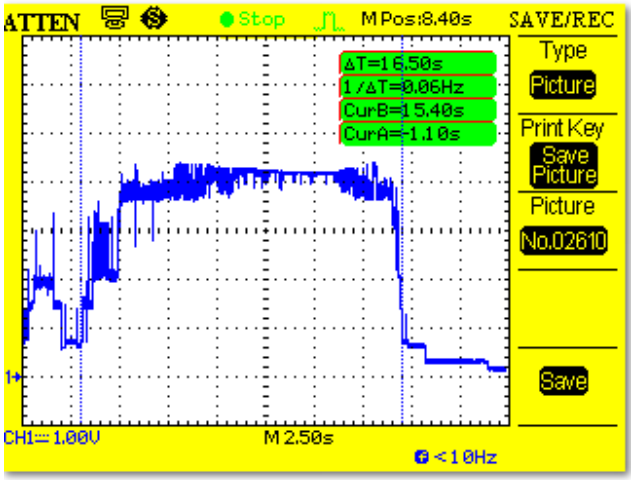


Fig. C.102.d. Proceso de subida de la foto al FTP

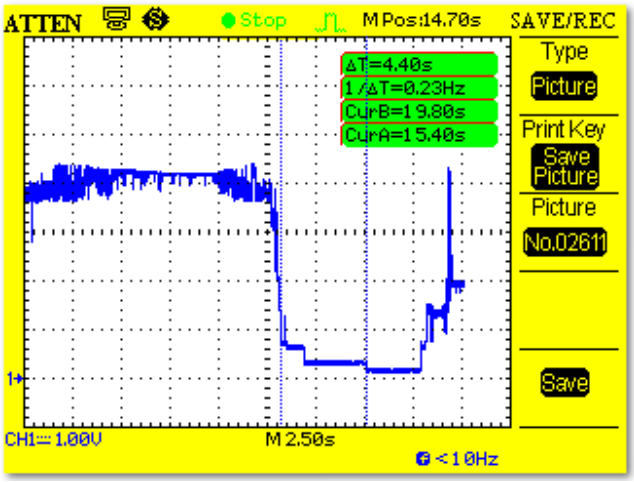


Fig. C.102.e. Proceso de echarse a dormir

El consumo del proceso total se muestra en la siguiente tabla:

	Duración total (ms)	Corriente total (mA)	Carga energetic total (mA·ms)
Proceso total	36000	205.418	7395072

Tabla C.108. Consumo del proceso total

C.3. Módulos de Comunicaciones.

C.3.1. Módulos XBee

Hay cuatro módulos XBee´s para analizar: 802.15.4, 868, DigiMesh y ZigBee.

Para este tipo de módulos, la estructura de los códigos difiere en las del resto. En el *loop()* del código, se hace el proceso completo desde que se enchufa el XBee hasta que se apaga porque, este tipo de módulos, requieren la realización de una serie de procesos cuando se enciende para que pueda transmitir y eso lleva tiempo. Si se dejara el módulo encendido en el *setup ()*, falsearía la medida, ya que ese tiempo previo antes del pico de envío, no aparecería.

C.3.1.1.- Módulo 802.15.4

El módulo 802.15.4 usa una serie de técnicas para evitar que todos los nodos empiecen emitiendo al mismo tiempo, entre ellas está el GTS (Guarantee Time Slots). Este sistema usa un nodo centralizado (coordinador) que da posiciones de tiempo para cada nodo de modo que cualquiera sabe cuando tiene que transmitir. Hay dieciséis posiciones de tiempo. Como primer paso, un nodo debe enviar al coordinador un mensaje de solicitud de GTS, como respuesta el coordinador enviará un mensaje conteniendo la posición asignada y el número de la posición asignada. Hay áreas de tramas especiales como los paquetes ACK que no requieren el método CSMA-CA para llevarse a cabo.

Para que se produzca la comunicación entre el módulo emisor y el receptor y que estén en la misma red, es necesario, previamente, configurar los módulos. Tienen que tener el mismo PAN ID y en el mismo canal de comunicación.

C.3.1.1.1.- Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo XBee 802.15.4 cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

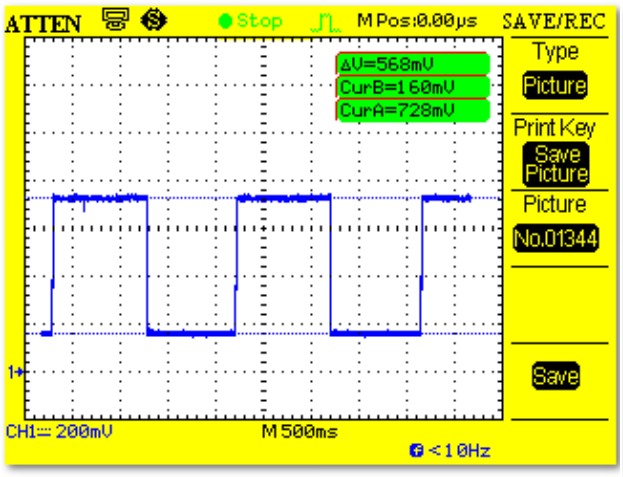


Fig. C.103.a. Medida general del proceso cada segundo

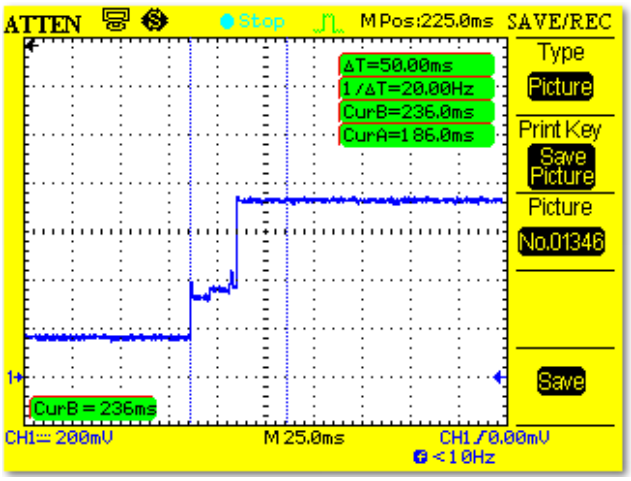


Fig. C.103.b. Proceso de encendido del módulo

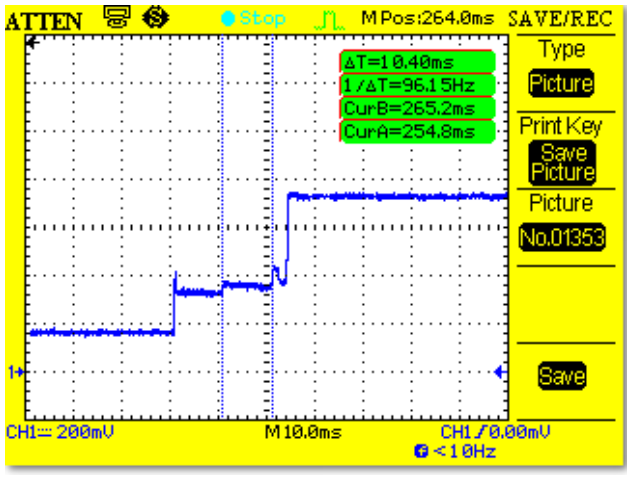


Fig. C.103.c. Zoom del proceso de encendido cada segundo

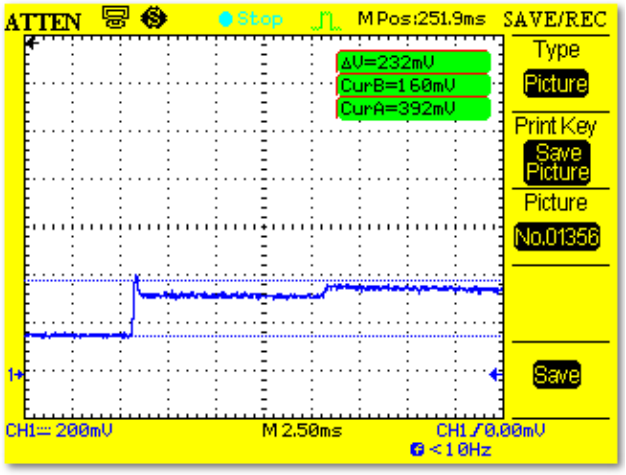


Fig. C.103.d. Más zoom en el proceso de encendido

Como se ve en la gráfica (b), el tiempo de ejecución del proceso de encendido del módulo es de 50 ms, mientras que el proceso de apagado se realiza instantáneamente.

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	50	39.24	1962.28
Estado On	1000	57.2	57200
Proceso Off	0	57.2	0

Tabla C.109. Consumo del proceso y estado on y del proceso off del módulo 802.15.4

C.3.1.1.2.- Envío de paquetes por unicast y en modo no cifrado

Este ejemplo muestra como enviar paquetes a un Gateway indicando la dirección MAC del módulo XBee receptor. El modo de envío se realizará por unicast y las tramas no estarán cifradas. La dirección MAC del receptor es 0013A2004071E4D2. Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 100 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

Para poder realizar este tipo de envíos, primeramente se debe configurar una serie de parámetros para los dos módulos, tanto para el emisor como para el receptor. Como se ha comentado anteriormente, los dos nodos tienen que estar en la misma red para que se establezca la comunicación. Por lo tanto, se configuraran con el mismo PAN ID y canal. Al enviarse el paquete sin cifrar, se desactivará el modo cifrado también, poniéndose a 0.

El PAN ID impuesto es 0x3332 y el canal el 0x0C.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

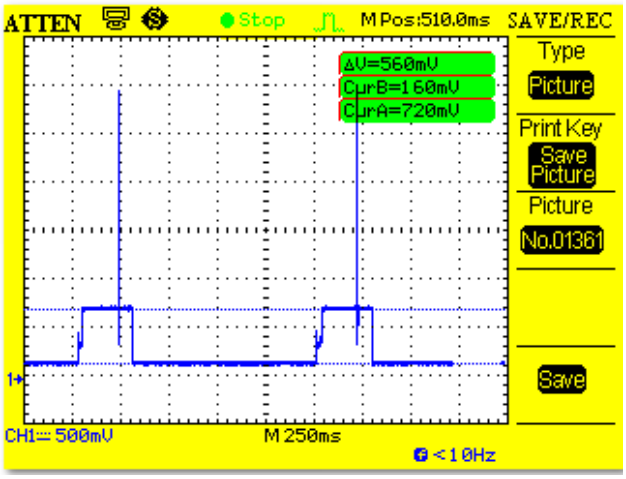


Fig. C.104.a. Medida general del proceso cada segundo

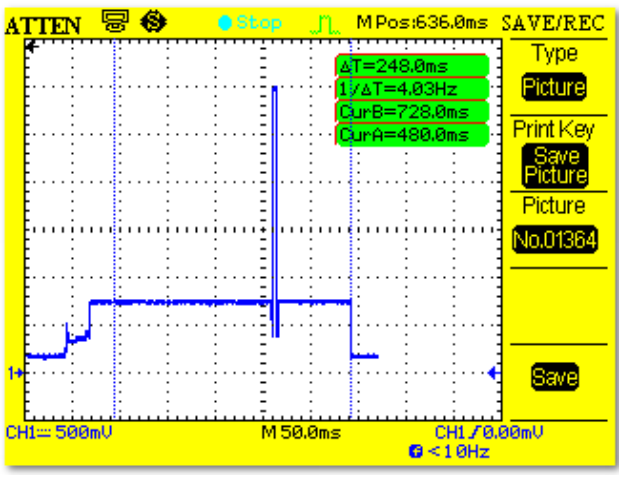


Fig. C.104.b. Proceso de transmisión y envío del paquete

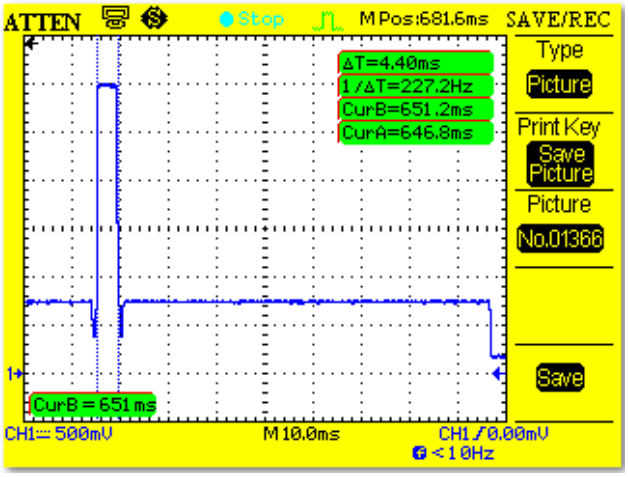


Fig. C.104.c. Tiempo de envío del paquete

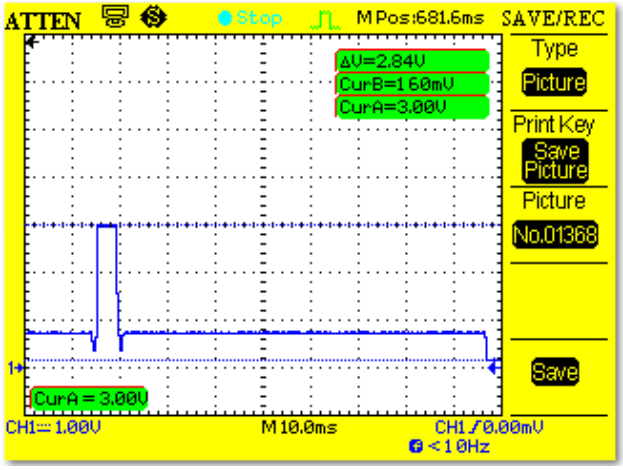


Fig. C.104.d. Consumo envío del paquete

Como se ve en la gráfica (b), durante el proceso de transmisión del paquete, hay un tiempo previo y posterior al envío del paquete.

El momento previo al envío es debido a la técnica GTS, explicada anteriormente y, también, a que los XBees tienen que escuchar el canal al que se ha conectado y, cuando comprueban que no hay nadie transmitiendo, se lanzan a enviar.

El momento posterior al envío del paquete corresponde al ACK. Todos los XBees y usando este modo de transmisión lo tienen. Es un paquete que el receptor envía al emisor como confirmación de que aquel ha recibido correctamente el paquete enviado.

En las gráficas (b) y (c), corresponde al envío del paquete. Normalmente, sale un pico con una diferencia de consumo considerable aunque el tiempo es mínimo.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	298	56.81	16931.56

Tabla C.110. Consumo del proceso total para el envío del paquete

C.3.1.1.3.- Envío de paquetes por broadcast y en modo no cifrado

Este ejemplo muestra como enviar paquetes con módulos XBee 802.15.4.

El modo de envío se realizará por broadcast y las tramas no estarán cifradas. La dirección broadcast (0x000000000000FFFF) se especifica como dirección de destino.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 100 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto y el canal tienen que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se desactivará, poniéndolo a 0.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

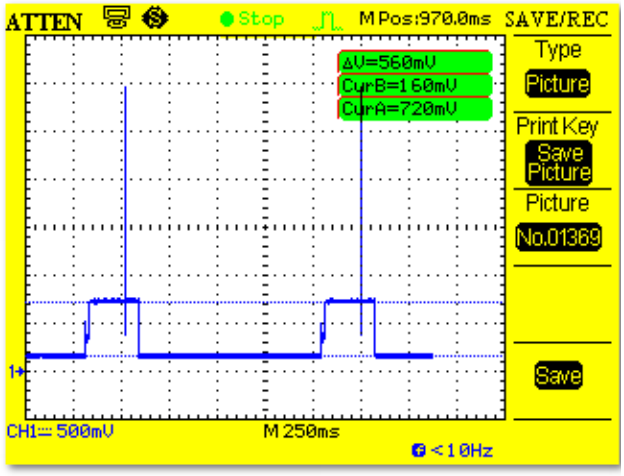


Fig. C.105.a. Medida general del proceso cada segundo

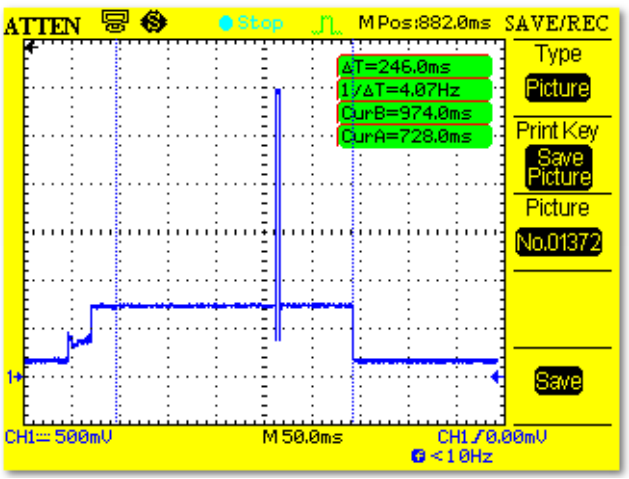


Fig. C.105.b. Proceso de transmisión y envío del paquete

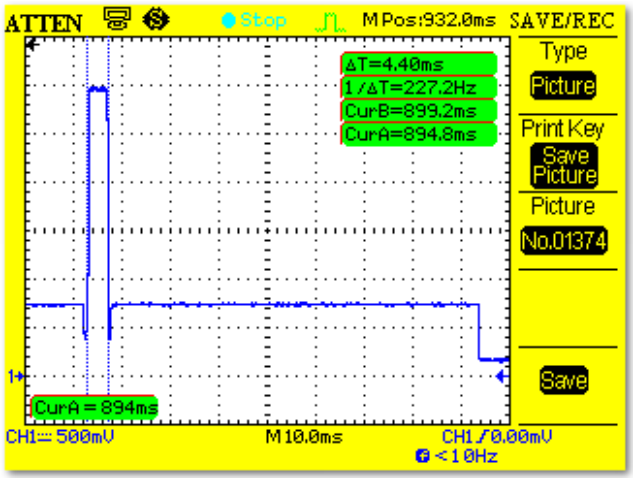


Fig. C.105.c. Tiempo de envío del paquete

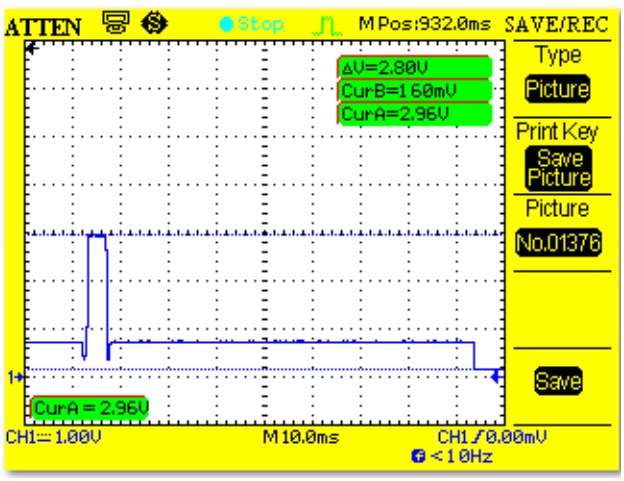


Fig. C.105.d. Consumo envío del paquete

Este tipo de modo de transmisión no tiene ACK por lo tanto el emisor no recibirá ningún mensaje de que la transmisión del paquete se ha realizado correctamente, por lo tanto, no hay seguridad de que el paquete les haya llegado a todos los nodos de la red.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	296	56.76	16801.16

Tabla C.111. Consumo del proceso total para el envío del paquete

C.3.1.1.4.- Envío de paquetes por unicast y en modo cifrado

Este ejemplo muestra como enviar paquetes a un Gateway indicando la dirección MAC del módulo XBee receptor. El modo de envío se realizará por unicast y las tramas estarán cifradas. La dirección MAC del receptor es 0013A2004071E4D2.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 94 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto y el canal tienen que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se activará, poniéndolo a 1.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

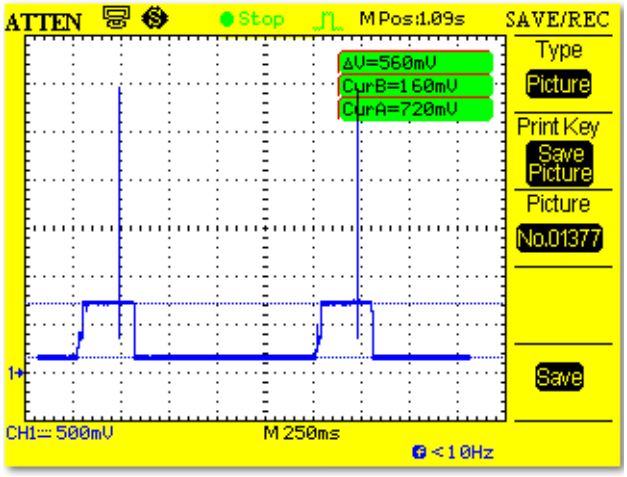


Fig. C.106.a. Medida general del proceso cada segundo

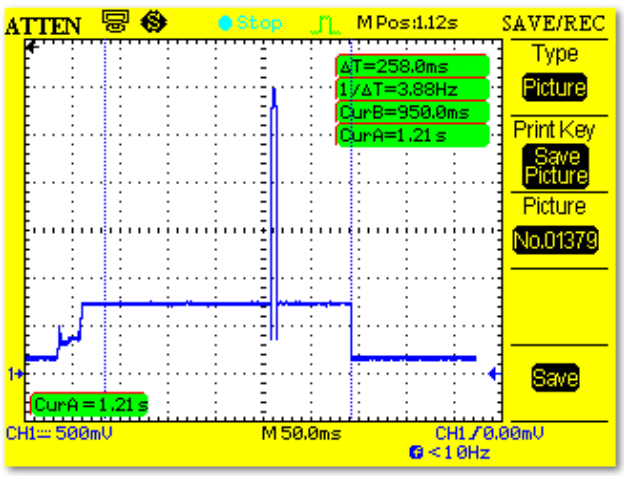


Fig. C.106.b. Proceso de transmisión y envío del paquete

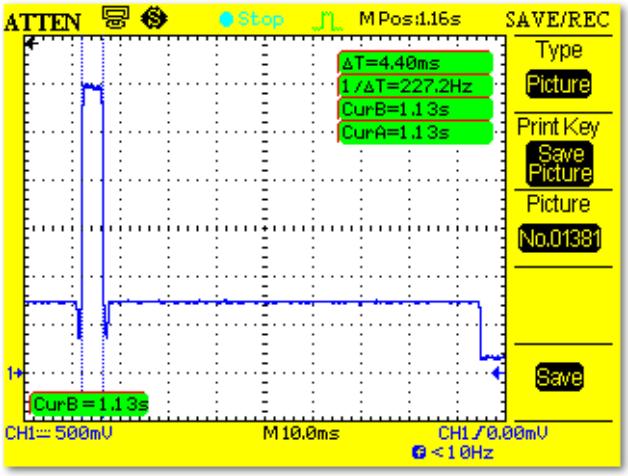


Fig. C.106.c. Tiempo de envío del paquete

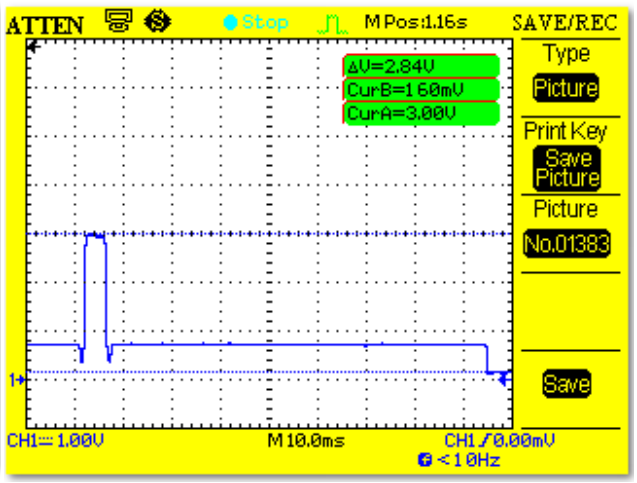


Fig. C.106.d. Consumo envío del paquete

Como se ve en la gráfica (b), el tiempo del proceso de transmisión del paquete es ligeramente mayor con respecto al anterior sin cifrar debido a que, esta vez, el envío de las tramas se realiza en modo cifrado.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	308	56.8	17495.56

Tabla C.112. Consumo del proceso total para el envío del paquete

C.3.1.1.5.- Envío de paquetes por broadcast y en modo cifrado

Este ejemplo muestra como enviar paquetes con módulos XBee 802.15.4.

El modo de envío se realizará por broadcast y las tramas estarán cifradas. La dirección broadcast (0x000000000000FFFF) se especifica como dirección de destino.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 95 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto y el canal tienen que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado desactivará, poniéndolo a 1.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

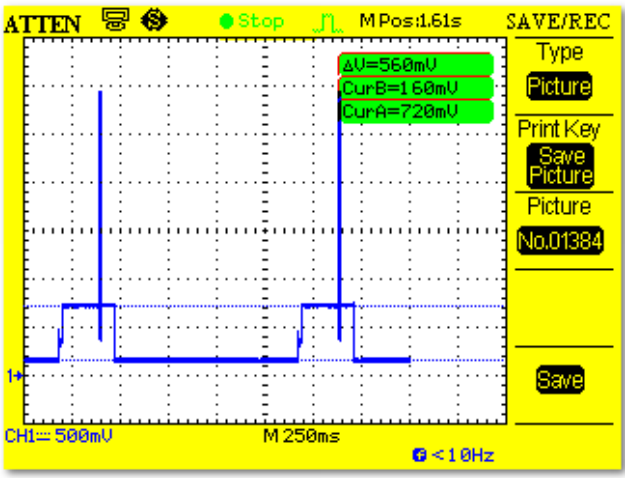


Fig. C.107.a. Medida general del proceso cada segundo

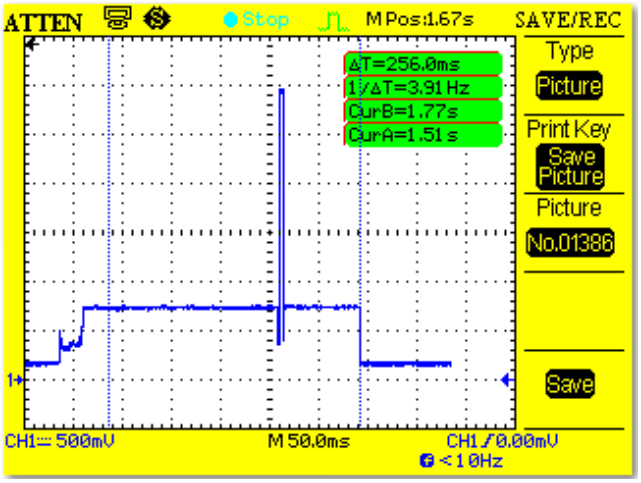


Fig. C.107.b. Proceso de transmisión y envío del paquete

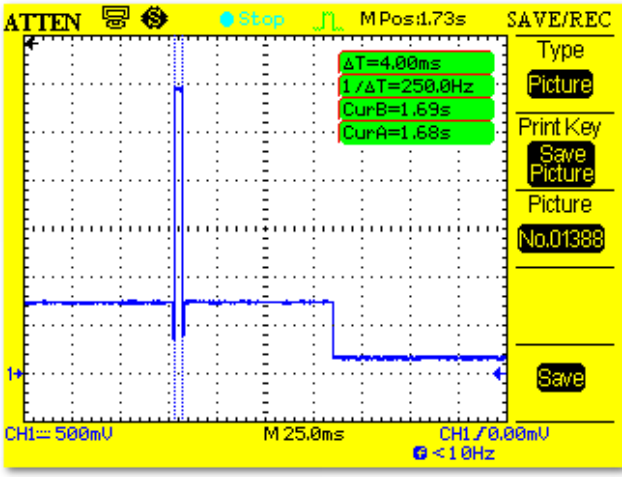


Fig. C.107.c. Tiempo de envío del paquete

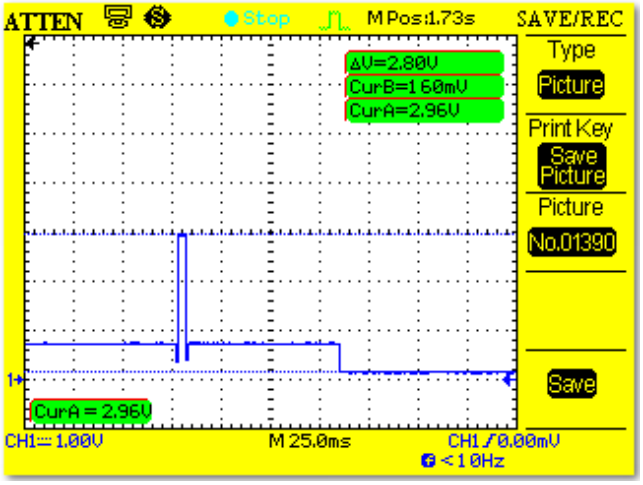


Fig. C.107.d. Consumo envío del paquete

Como se ve en la gráfica (b), el tiempo del proceso de transmisión del paquete es ligeramente mayor con respecto al anterior sin cifrar debido a que, esta vez, el envío de las tramas se realiza en modo cifrado.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	306	56.748	17365.16

Tabla C.113. Consumo del proceso total para el envío del paquete

C.3.1.1.6.- Comandos AT

Este ejemplo muestra como configurar los parámetros XBee básicos para comunicarse entre diferentes dispositivos XBee usando la misma red.

Para leer los comandos AT de configuración del XBee en necesario encender el módulo previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

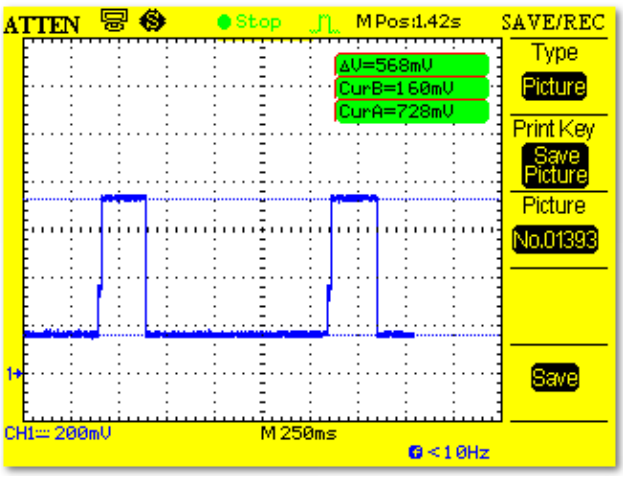


Fig. C.108.a. Medida general del proceso cada segundo

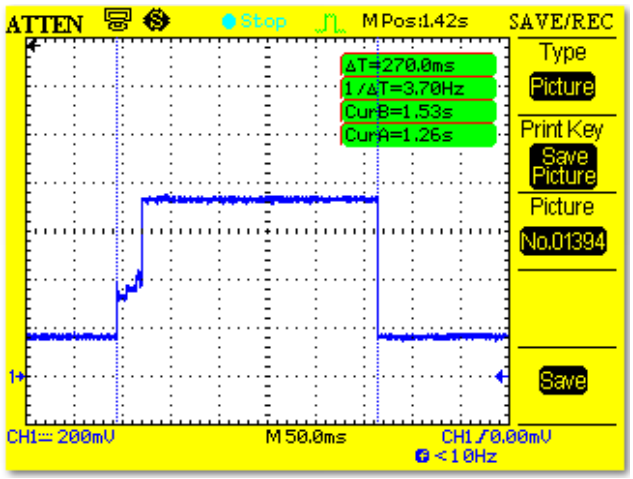


Fig. C.108.b. Tiempo de ejecución del proceso total

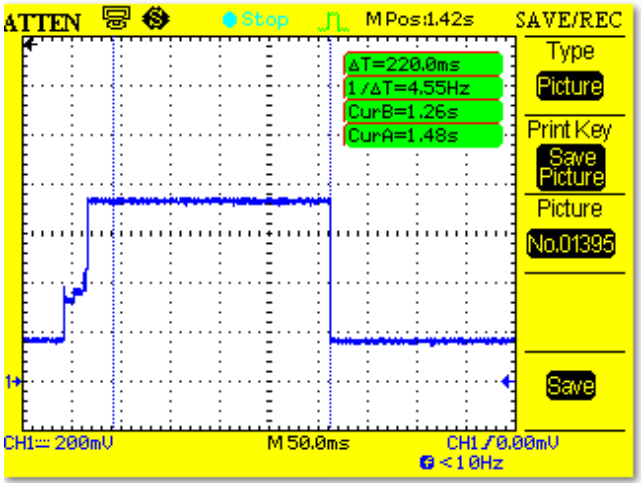


Fig. C.108.c. Tiempo de ejecución del proceso total

Como se ve en la gráfica (c), el tiempo de ejecución de los comandos AT es de 220 ms, aunque no consumen nada cuando se enciende el módulo.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	270	53.87	14546.28

Tabla C.114. Consumo del proceso total para el envío del paquete

C.3.1.1.7. - Obtención del RSSI

Este ejemplo muestra cómo conseguir el valor del RSSI del último paquete recibido. En este protocolo, es posible extraer esta información de cada paquete. Al lado, hay una función API que muestra esta información. Antes de correr este ejemplo, hay que asegurarse de que hay otro emisor enviando paquetes al módulo XBee para recibir información.

Para que el valor del RSSI llegue correctamente, es necesario dejar encendido el XBee.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

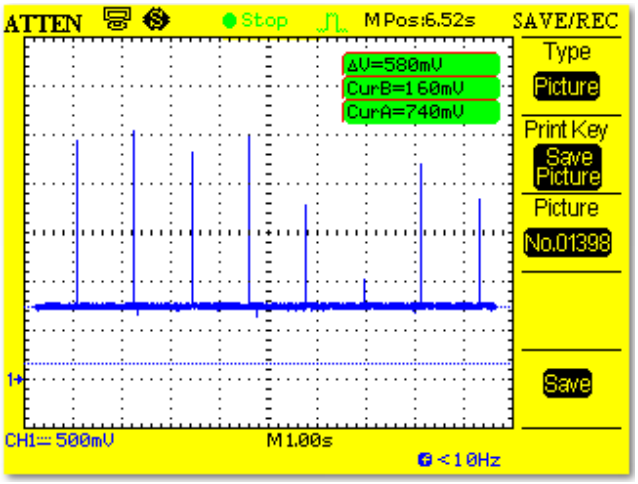


Fig. C.109.a. Medida general del proceso cada 5 segundos

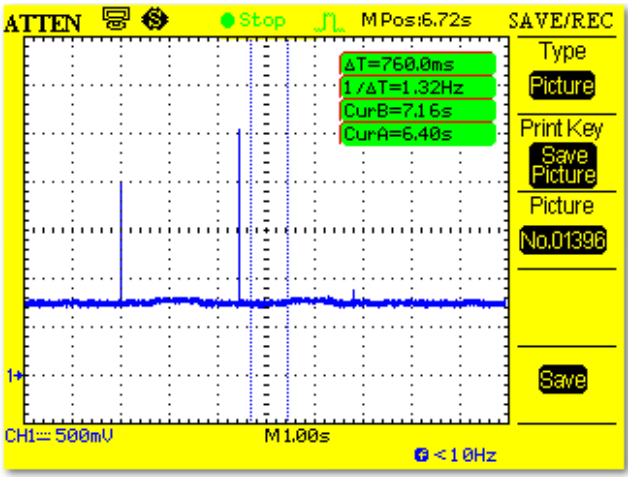


Fig. C.109.b. Tiempo de ejecución del proceso

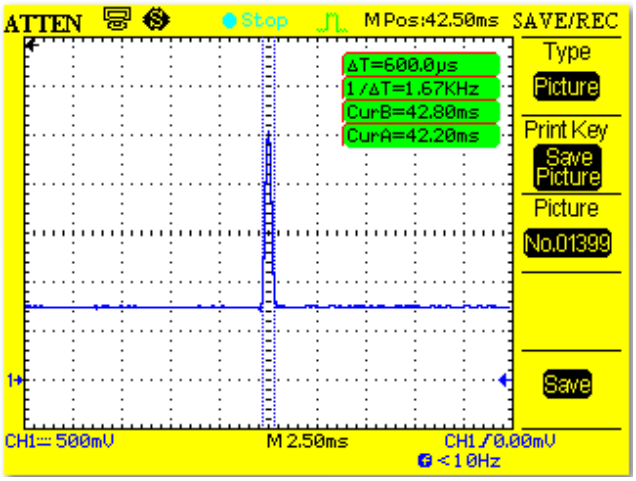


Fig. C.109.c. Tiempo de la señal RSSI

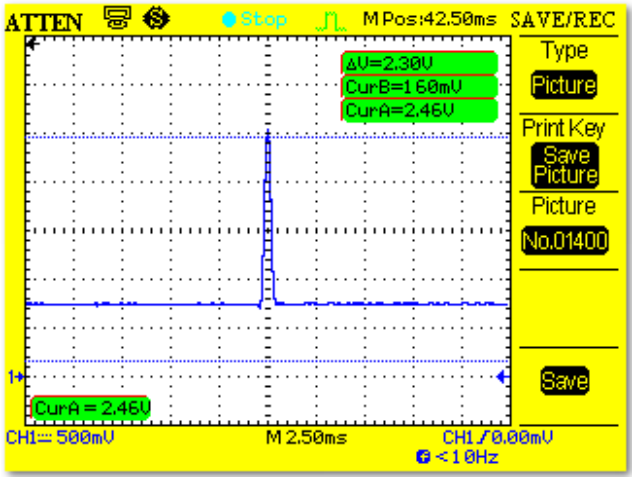


Fig. C.109.d. Consumo señal RSSI

Como se ve en la gráfica (a), la señal sale totalmente plana dificultando saber cuánto es el tiempo de ejecución del proceso. Para ello, se añadió una serie de funciones al código que produce un consumo extra a lo ejecutado y, así, distinguir mejor el tiempo de ejecución de este proceso, como se ve claramente en la gráfica (b), aunque el consumo extra después de encender el módulo es 0.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	760	0	0

Tabla C.115. Consumo del proceso de obtención del RSSI

C.3.1.2.- Módulo 868

Para que se produzca la comunicación entre el módulo emisor y el receptor y que estén en la misma red, es necesario, previamente, configurar los módulos.

Para este tipo de módulos, solo es necesario que los dos tengan el mismo PAN ID, ya que el 868 trabaja en un solo canal y no es necesario configurarlo para que todos los módulos XBee 868 utilicen el mismo.

C.3.1.2.1.- Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo XBee 868 cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

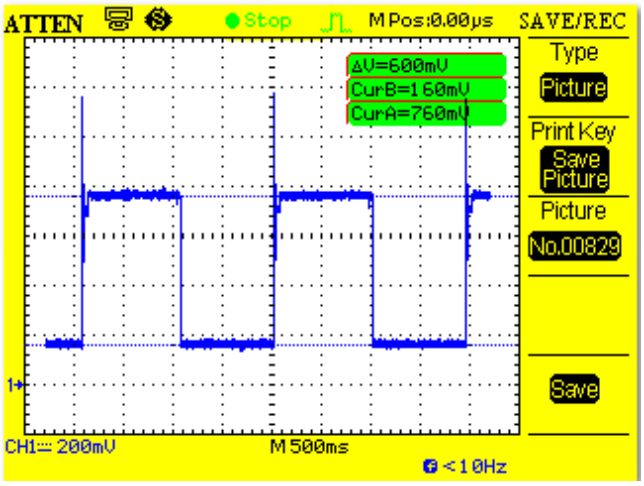


Fig. C.110.a. Medida general del proceso cada segundo

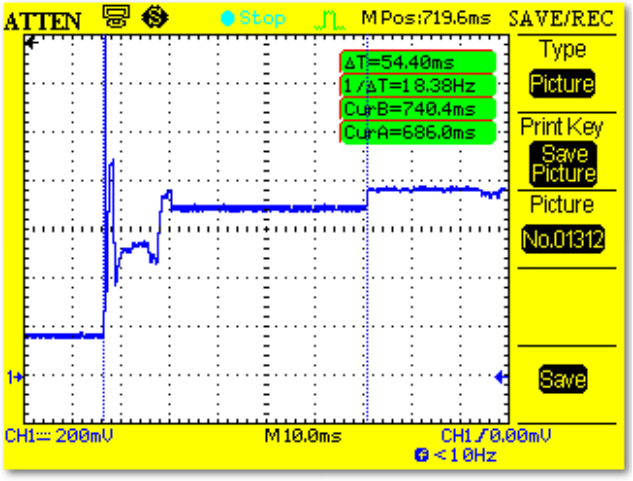


Fig. C.110.b. Proceso de encendido del módulo

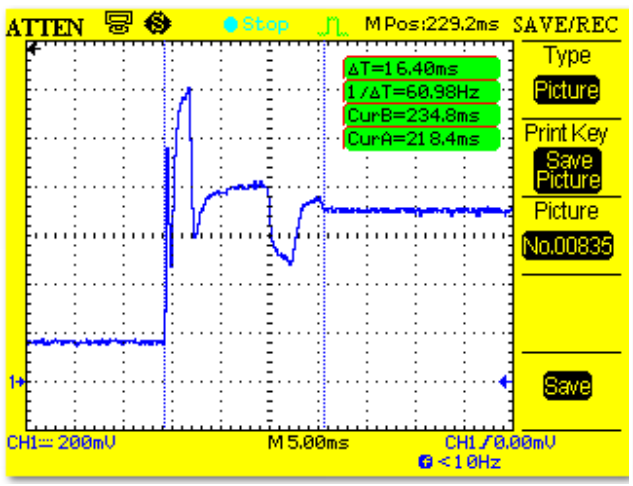


Fig. C.110.c. Pico de encendido

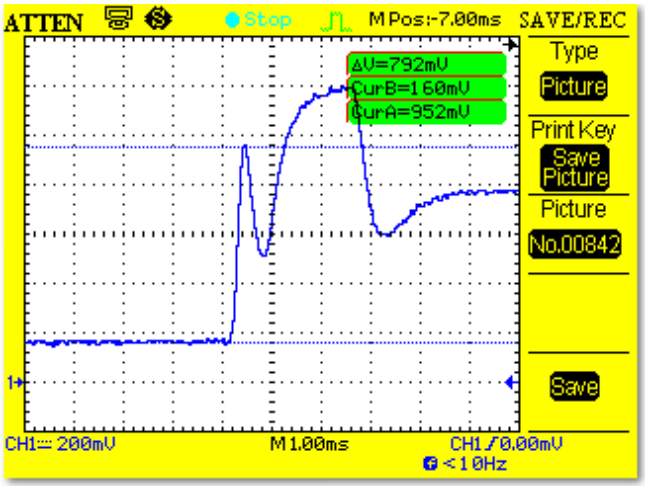


Fig. C.110.d. Zoom del pico de encendido

Como se ve en la gráfica (b), el tiempo de ejecución del proceso de encendido del módulo es de 54.4 ms, mientras que el proceso de apagado se realiza instantáneamente.

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	54.4	55.1	2997.448
Estado On	1000	60.4	60400
Proceso Off	0	60.4	0

Tabla C.116. Consumo del proceso y estado on y del proceso off del módulo 868

C.3.1.2.2.- Envío de paquetes por unicast y en modo no cifrado

Este ejemplo muestra como enviar paquetes a un Gateway indicando la dirección MAC del módulo XBee receptor. El modo de envío se realizará por unicast y las tramas no estarán cifradas. La dirección MAC del receptor es 0013A20040894745.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 100 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

Para poder realizar este tipo de envíos, primeramente se debe configurar una serie de parámetros para los dos módulos, tanto para el emisor como para el receptor. Como se ha comentado anteriormente, los dos nodos tienen que estar en la misma red para que se establezca la comunicación. Por lo tanto, se configuraran con el mismo PAN ID. Al enviarse el paquete sin cifrar, se desactivará el modo cifrado también, poniéndose a 0.

El PAN ID impuesto es 0x7FFF.

En algunas ocasiones, este módulo, cuando no envía correctamente el paquete a la primera, tiene reintentos hasta conseguirlo, mostrándose dos o tres picos de envíos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado, **enviando a la primera**:

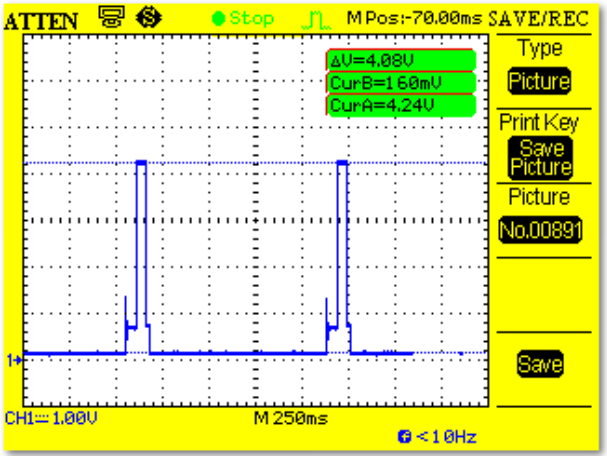


Fig. C.111.a. Medida general del proceso cada segundo

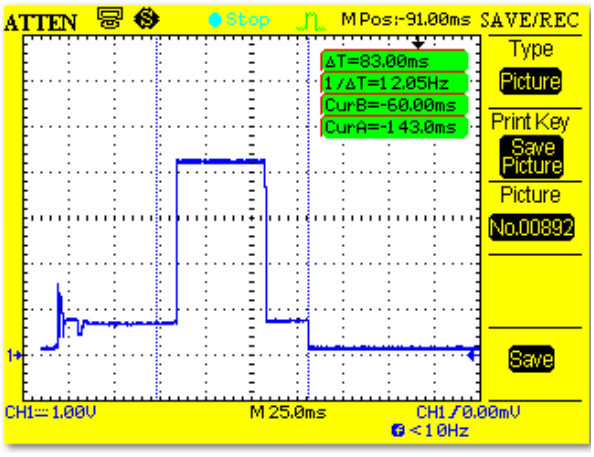


Fig. C.111.b. Proceso de transmisión y envío del paquete

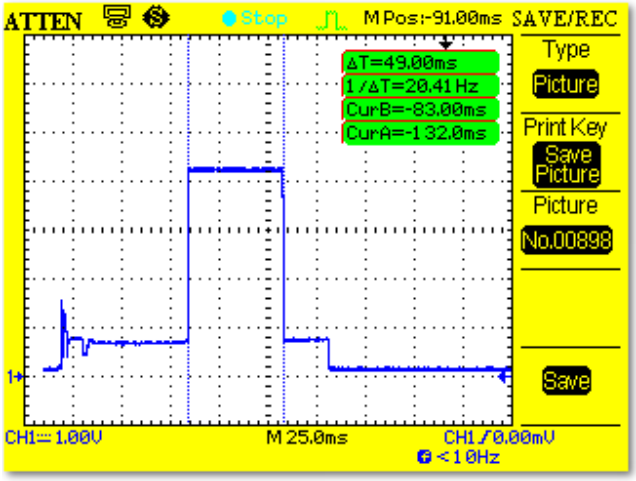


Fig. C.111.c. Tiempo de envío del paquete

Como se ve en la gráfica (b), durante el proceso de transmisión del paquete, hay un tiempo previo y posterior al envío del paquete.

Se observa que el tiempo del momento previo al envío del paquete es muchísimo menor al del módulo anterior, ya que este módulo no usa la técnica que usaba el otro.

El momento posterior al envío del paquete corresponde al ACK. Todos los XBees y usando este modo de transmisión lo tienen. Es un paquete que el receptor envía al emisor como confirmación de que aquel ha recibido correctamente el paquete enviado.

En las gráficas (b) y (c), corresponde al envío del paquete. Normalmente, sale un pico con una diferencia de consumo considerable.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	137	182.93	25062.648

Tabla C.117. Consumo del proceso total para el envío del paquete

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado, **con reintentos** en el envío:

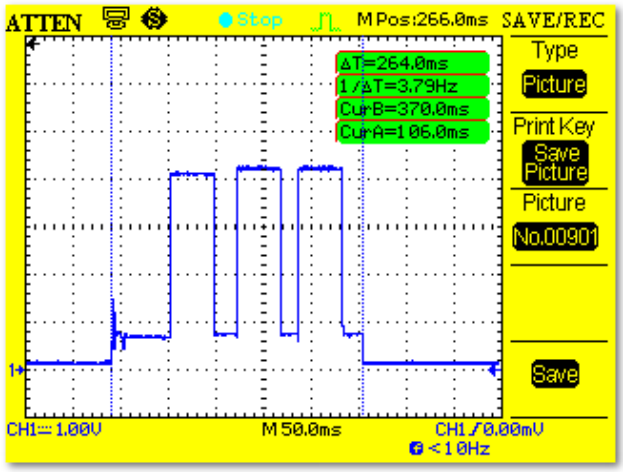


Fig. C.112.a. Tiempo de ejecución del proceso total

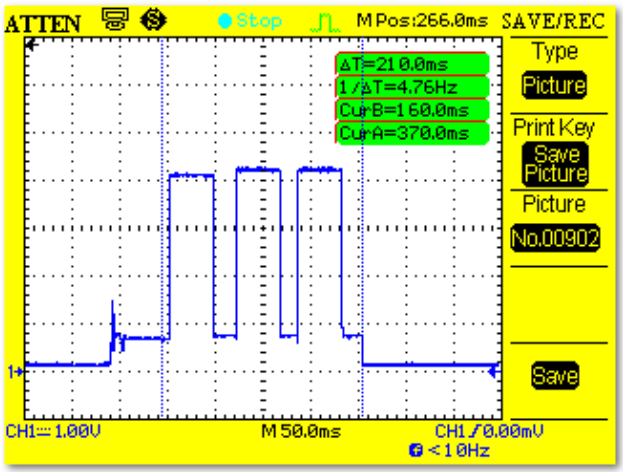


Fig. C.112.b. Proceso de transmisión y reintentos

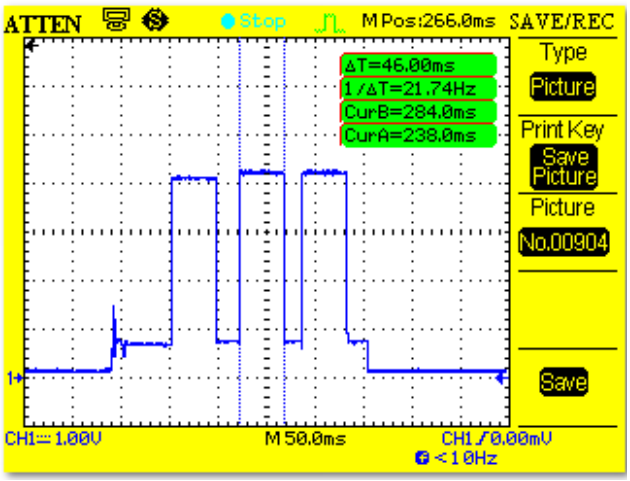


Fig. C.112.c. Tiempo de ejecución de un reintento

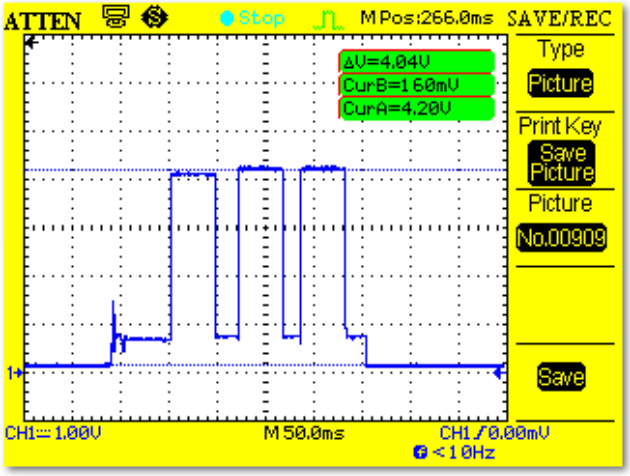


Fig. C.112.d. Consumo en el segundo y tercer reintento

Como se observa en las gráficas, se envía el paquete hasta tres veces para que lo reciba el módulo receptor.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	264	237.12	62601.44

Tabla C.118. Consumo del proceso total para el envío del paquete

C.3.1.2.3.- Envío de paquetes por broadcast y en modo no cifrado

Este ejemplo muestra como enviar paquetes con módulos XBee 868.

El modo de envío se realizará por broadcast y las tramas no estarán cifradas. La dirección

broadcast (0x000000000000FFFF) se especifica como dirección de destino.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 100 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto tiene que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se desactivará, poniéndolo a 0.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

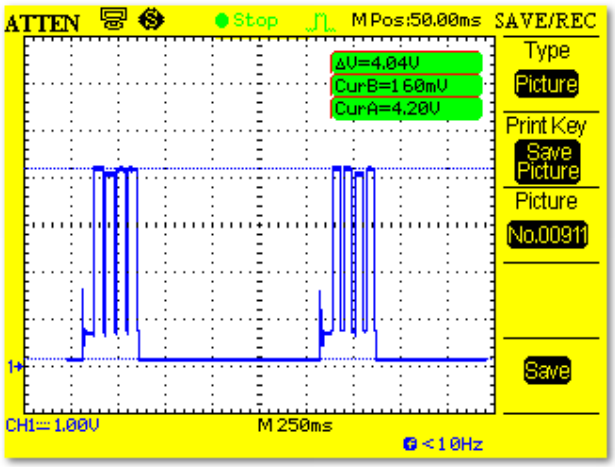


Fig. C.113.a. Medida general del proceso cada segundo

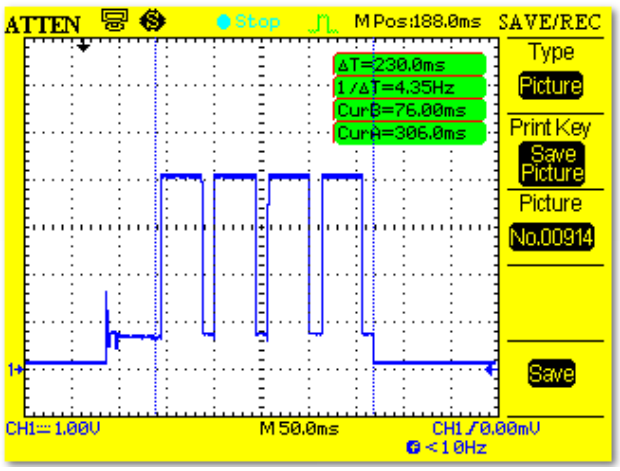


Fig. C.113.b. Proceso de transmisión y envío del paquete

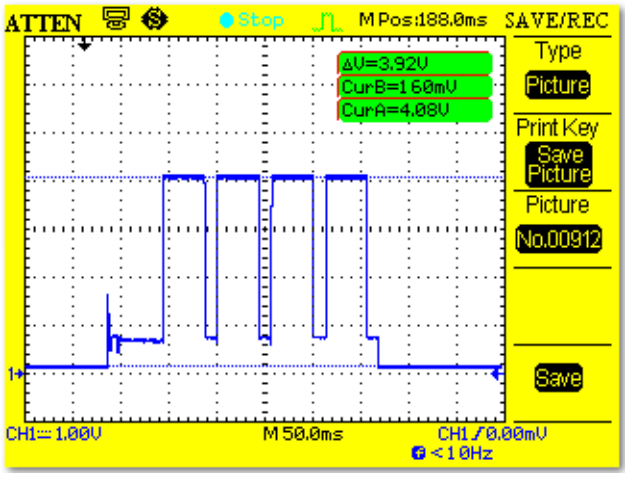


Fig. C.113.c. Consumo envío del paquete

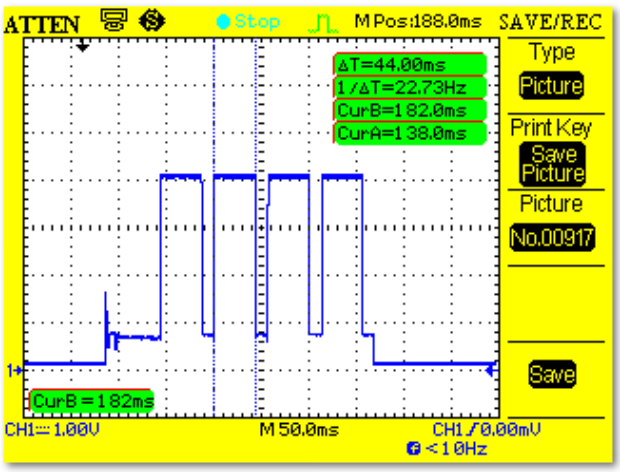


Fig. C.113.d. Tiempo de cada reintento

Como se observa en las gráficas, hay cuatro picos de envíos de datos. Esto es debido a que le comando MT, que define el número de transmisiones de difusión adicionales, por defecto, es tres. Como todos los paquetes de difusión se transmiten MT+1 veces para asegurarse de que se recibe, entonces se producen cuatro transmisiones, que son los cuatr picos que se ven ahí.

Este tipo de modo de transmisión no tiene ACK por lo tanto el emisor no recibirá ningún mensaje de que la transmisión del paquete se ha realizado correctamente, por lo tanto, no hay seguridad de que el paquete les haya llegado a todos los nodos de la red.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	282	266.66	75200.64

Tabla C.119. Consumo del proceso total para el envío del paquete

C.3.1.2.4.- Envío de paquetes por unicast y en modo cifrado

Este ejemplo muestra como enviar paquetes a un Gateway indicando la dirección MAC del módulo XBee receptor. El modo de envío se realizará por unicast y las tramas estarán cifradas. La dirección MAC del receptor es 0013A20040894745.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 100 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto tiene que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se activará, poniéndolo a 1.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

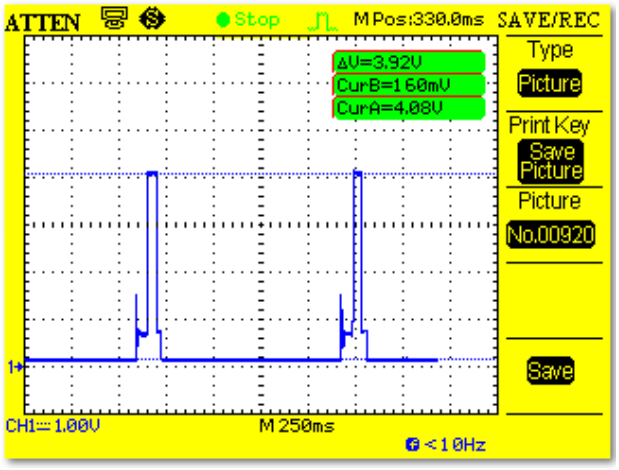


Fig. C.114.a. Medida general del proceso cada segundo

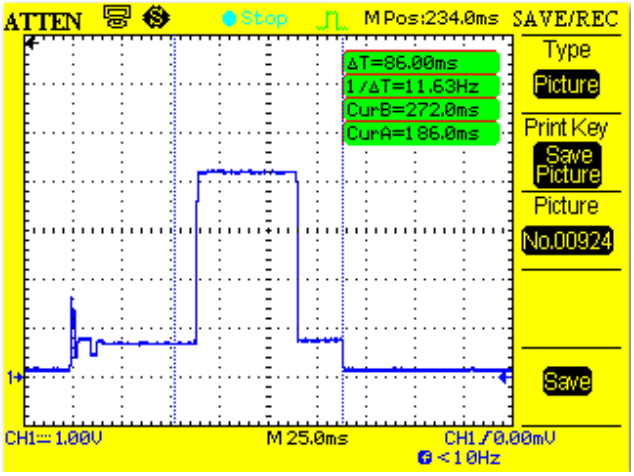


Fig. C.114.b. Proceso de transmisión y envío del paquete

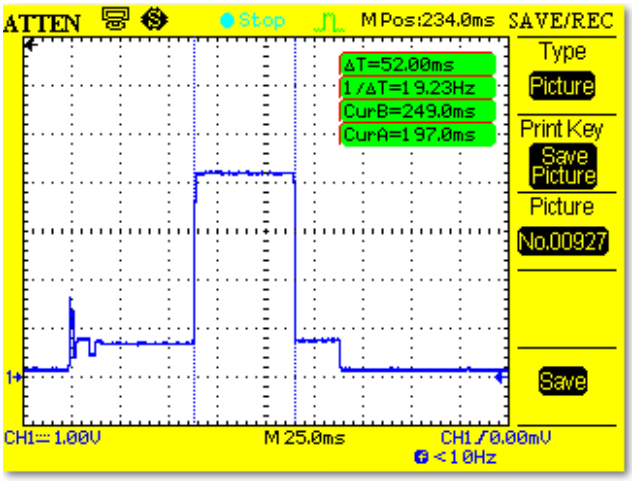


Fig. C.114.c. Tiempo de envío del paquete

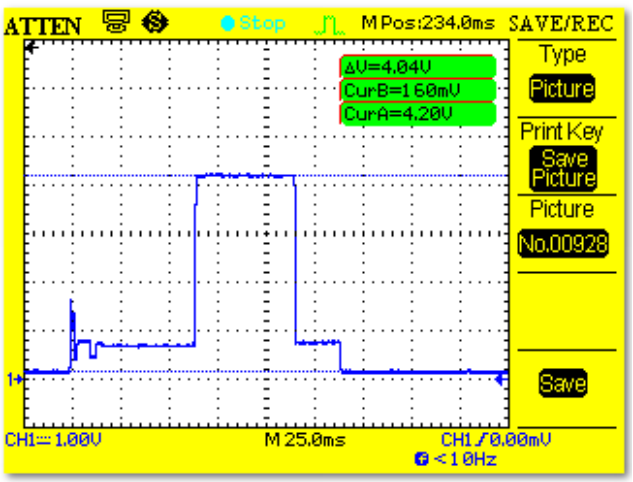


Fig. C.114.d. Consumo envío del paquete

Como se ve en la gráfica (b), el tiempo del proceso de transmisión del paquete es ligeramente mayor con respecto al de sin cifrar debido a que, esta vez, el envío de las tramas se realiza en modo cifrado.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	140	186.28	26079.84

Tabla C.120. Consumo del proceso total para el envío del paquete

C.3.1.2.5.- Envío de paquetes por broadcast y en modo cifrado

Este ejemplo muestra como enviar paquetes con módulos XBee 868.

El modo de envío se realizará por broadcast y las tramas estarán cifradas. La dirección broadcast (0x000000000000FFFF) se especifica como dirección de destino.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 100 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto tiene que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se activará, poniéndolo a 1.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

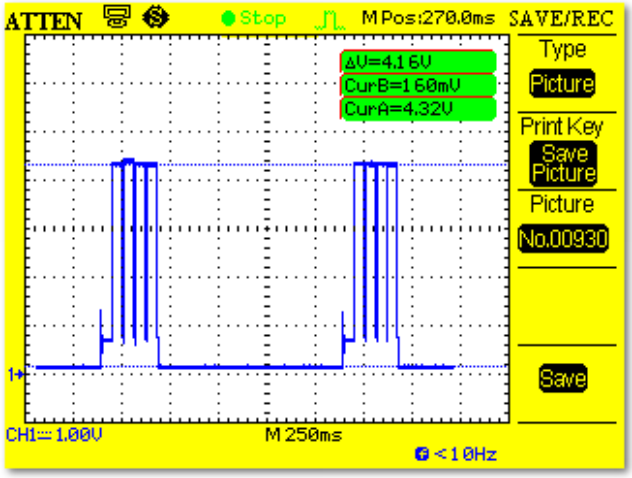


Fig. C.115.a. Medida general del proceso cada segundo

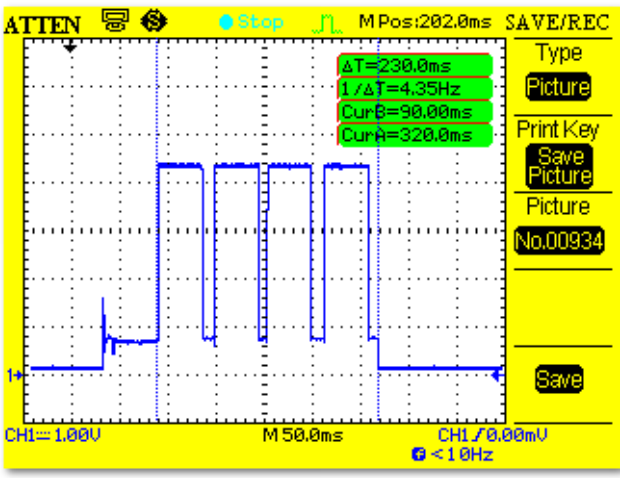


Fig. C.115.b. Proceso de transmisión y envío del paquete

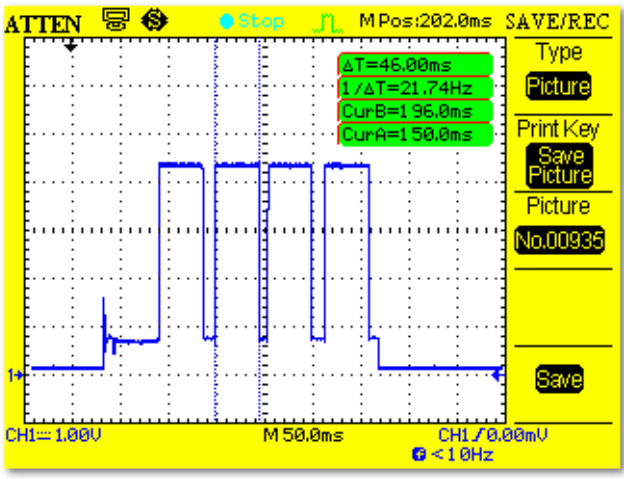


Fig. C.115.c. Tiempo de ejecución de cada reintento

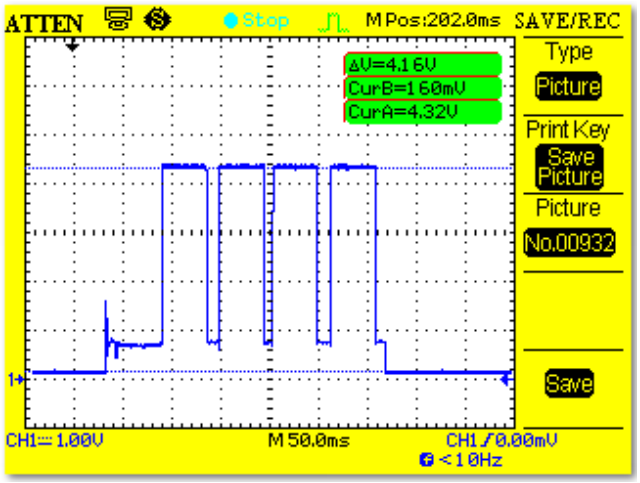


Fig. C.115.d. Consumo envío del paquete

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	286	288.51	82514.24

Tabla C.121. Consumo del proceso total para el envío del paquete

C.3.1.2.6.- Comandos AT

Este ejemplo muestra como configurar los parámetros XBee básicos para comunicarse entre diferentes dispositivos XBee usando la misma red.

Para leer los comandos AT de configuración del XBee en necesario encender el módulo previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

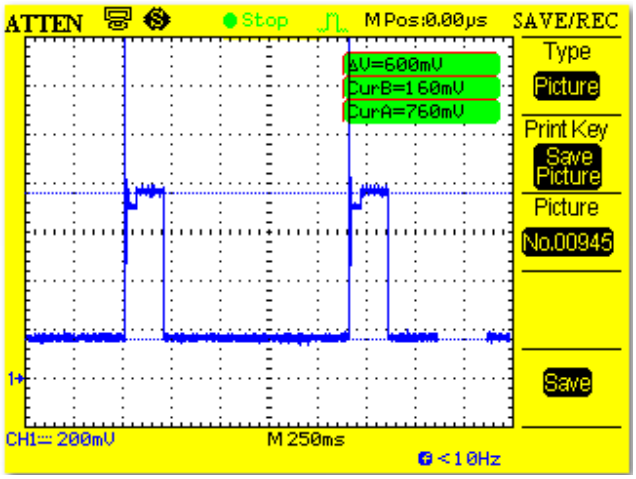


Fig. C.116.a. Medida general del proceso cada segundo

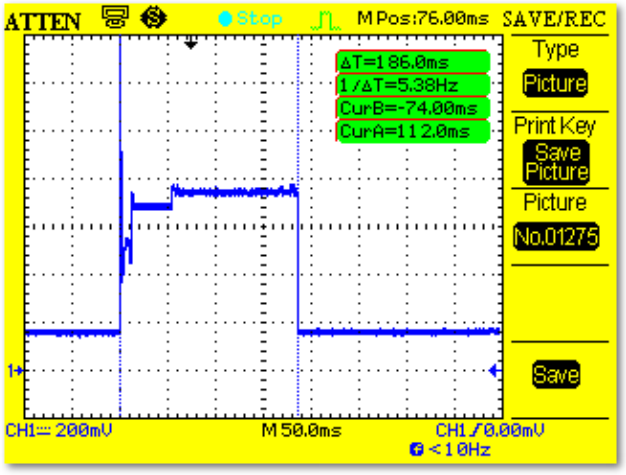


Fig. C.116.b. Tiempo de ejecución del proceso

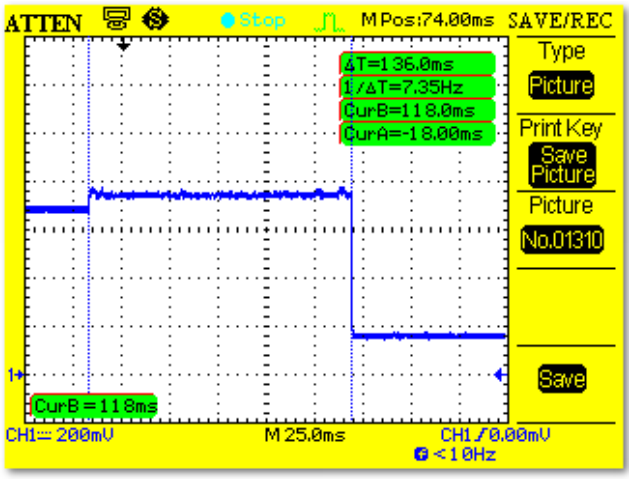


Fig. C.116.c. Tiempo de ejecución de los parámetros

Como se ve en la gráfica (c), el tiempo de ejecución de los comandos AT es de 136 ms, aunque no consumen nada cuando se enciende el módulo.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	186	60.27	11211.84

Tabla C.122. Consumo del proceso total de la configuración de los comandos AT

C.3.1.2.7. - Obtención del RSSI

Este ejemplo muestra cómo conseguir el valor del RSSI del último paquete recibido. En este

protocolo, es posible extraer esta información de cada paquete. Al lado, hay una función API que muestra esta información. Antes de correr este ejemplo, hay que asegurarse de que hay otro emisor enviando paquetes al módulo XBee para recibir información.

Para que el valor del RSSI llegue correctamente, es necesario dejar encendido el XBee.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

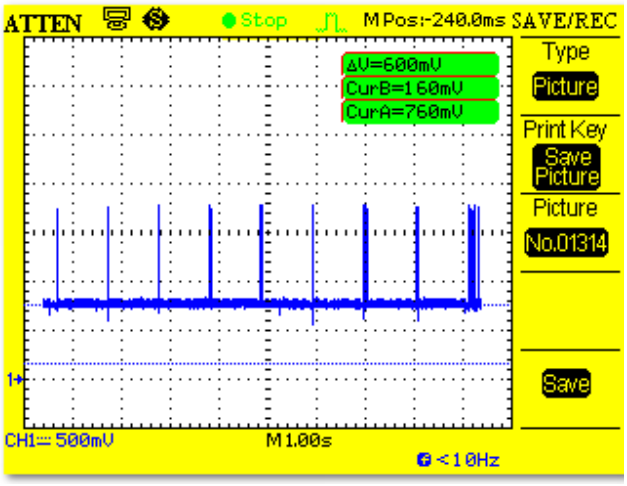


Fig. C.117.a. Medida general del proceso cada segundo

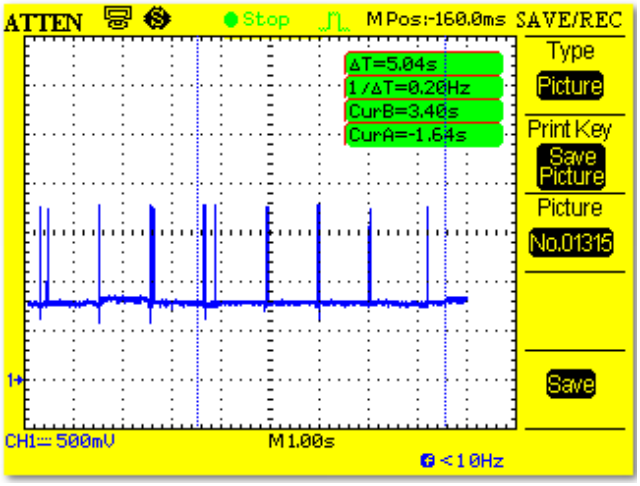


Fig. C.117.b. Tiempo de ejecución del proceso

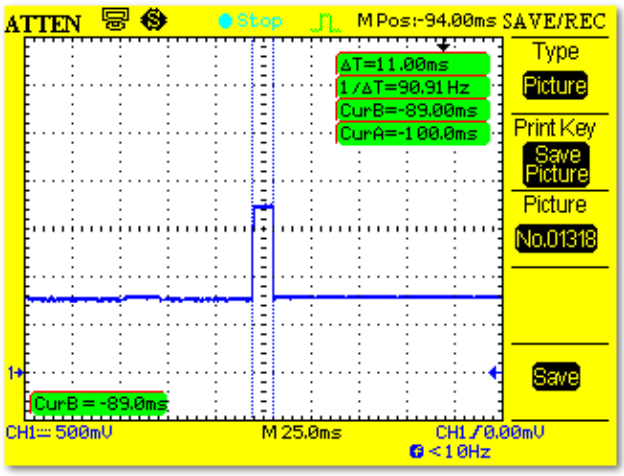


Fig. C.117.c. Tiempo de ejecución de cada pulso

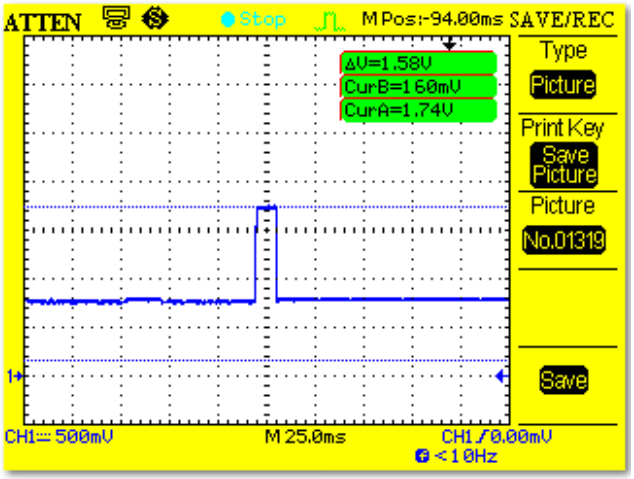


Fig. C.117.d. Consumo del pulso

Como se ve en la gráfica (a), la señal sale totalmente plana dificultando saber cuánto es el tiempo de ejecución del proceso. Para ello, se añadió una serie de funciones al código que produce un consumo extra a lo ejecutado y, así, distinguir mejor el tiempo de ejecución de este proceso, como se ve claramente en la gráfica (b).

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	5040	1.283	6468

Tabla C.123. Consumo del proceso de obtención del RSSI

C.3.1.3.- Módulo DigiMesh

Para que se produzca la comunicación entre el módulo emisor y el receptor y que estén en la misma red, es necesario, previamente, configurar los módulos.

Para este tipo de módulos, es necesario que los dos tengan el mismo PAN ID y estén en el mismo canal.

C.3.1.3.1.- Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo DM cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

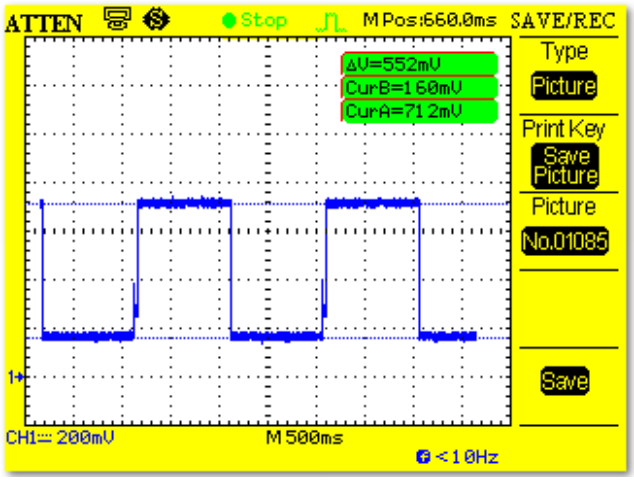


Fig. C.118.a. Medida general del proceso cada segundo

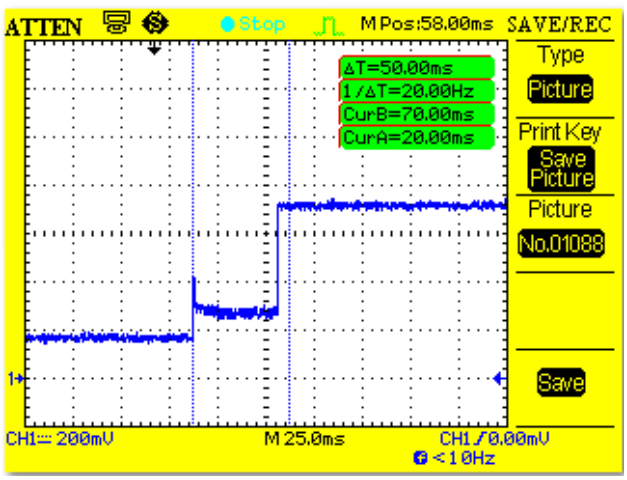


Fig. C.118.b. Proceso de encendido del módulo

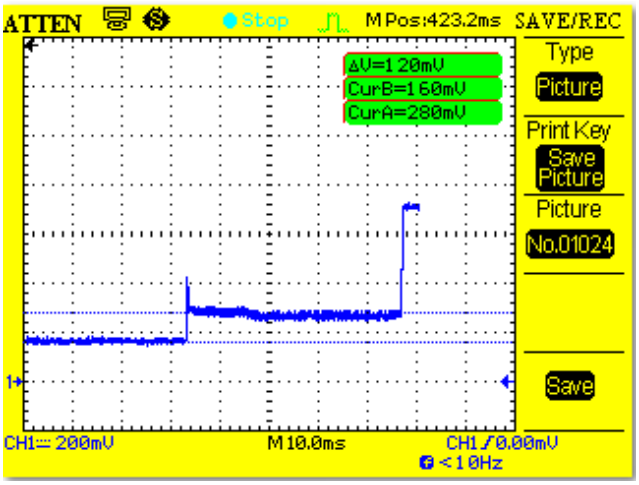


Fig. C.118.c. Zoom proceso de encendido

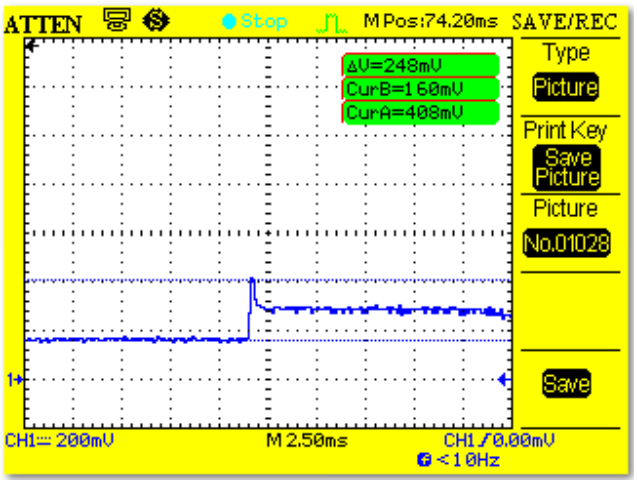


Fig. C.118.d. Pico del proceso de encendido

Como se ve en la gráfica (b), el tiempo de ejecución del proceso de encendido del módulo es de 50 ms, mientras que el proceso de apagado se realiza instantáneamente.

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	50	17.24	862.24
Estado On	1000	55.6	55600
Proceso Off	0	55.6	0

Tabla C.124. Consumo del proceso y estado on y del proceso off del módulo DigiMesh

C.3.1.3.2.- Envío de paquetes por unicast y en modo no cifrado

Este ejemplo muestra como enviar paquetes a un Gateway indicando la dirección MAC del módulo XBee receptor. El modo de envío se realizará por unicast y las tramas no estarán cifradas. La dirección MAC del receptor es 0013A200406C16CE.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 73 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

Para poder realizar este tipo de envíos, primeramente se debe configurar una serie de parámetros para los dos módulos, tanto para el emisor como para el receptor. Como se ha comentado anteriormente, los dos nodos tienen que estar en la misma red para que se establezca la comunicación. Por lo tanto, se configuraran con el mismo PAN ID y estarán en el mismo canal. Al enviarse el paquete sin cifrar, se desactivará el modo cifrado también, poniéndose a 0.

El PAN ID impuesto es 0x7FFF y el canal el 0x10.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

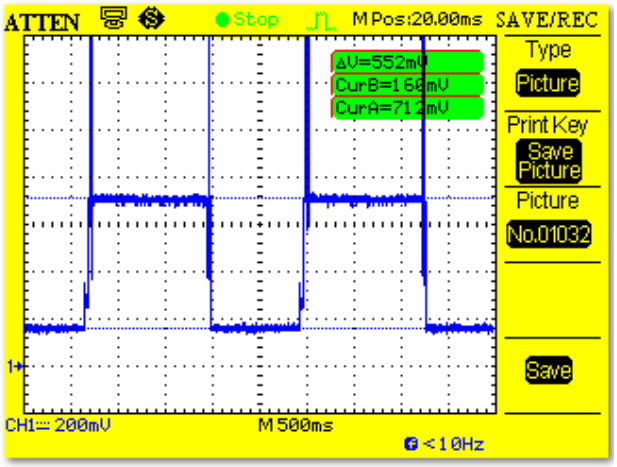


Fig. C.119.a. Medida general del proceso cada segundo

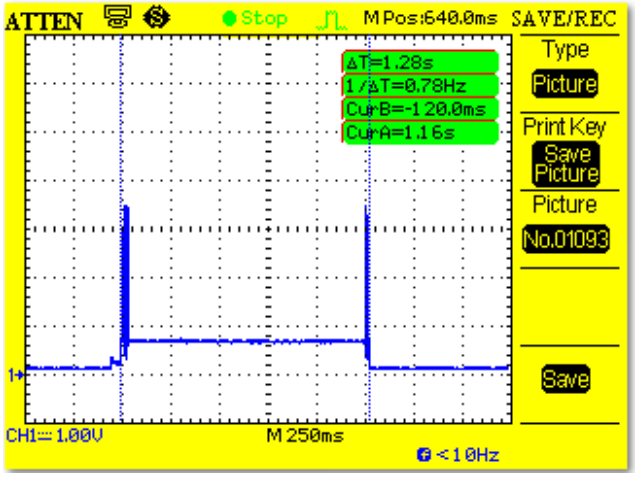


Fig. C.119.b. Proceso de transmisión y envío del paquete

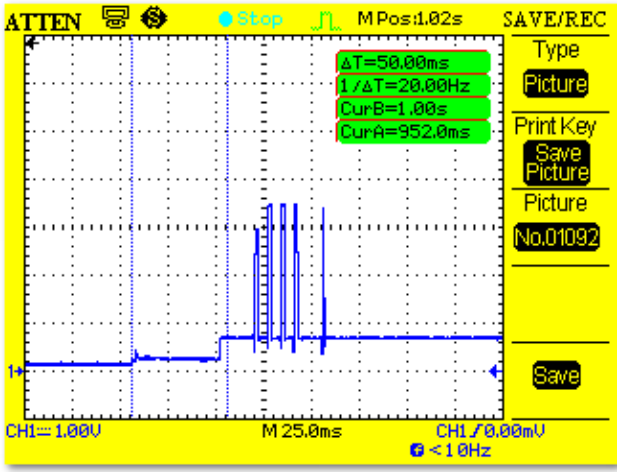


Fig. C.119.c. Proceso de encendido del módulo

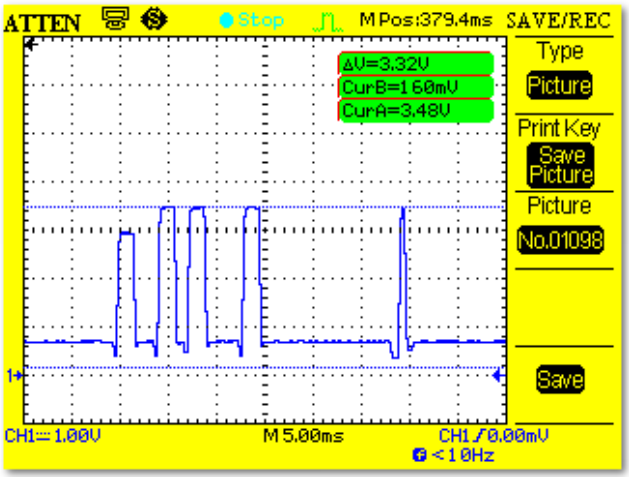


Fig. C.119.d. Consumo picos producidos al principio de la transmisión

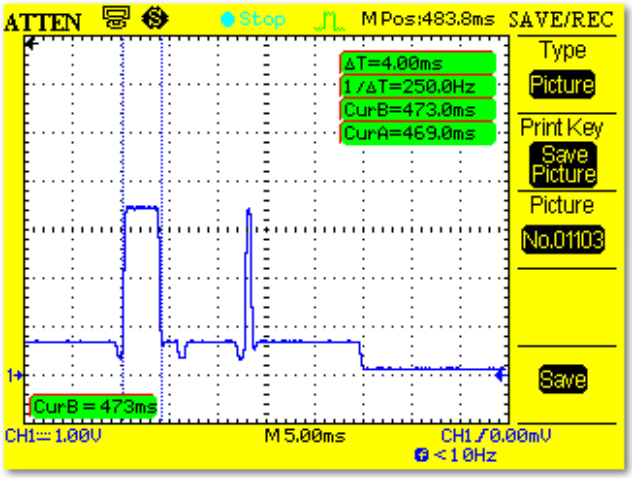


Fig. C.119.e. Pico del envío del paquete

Como se ve en la gráfica (b), durante el proceso de transmisión del paquete, hay un tiempo previo y posterior al envío del paquete.

El momento posterior al envío del paquete corresponde al ACK. Todos los XBees y usando este modo de transmisión lo tienen. Es un paquete que el receptor envía al emisor como confirmación de que aquel ha recibido correctamente el paquete enviado.

La gráfica (e), corresponde al envío del paquete. Normalmente, sale un pico con una diferencia de consumo considerable.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	1324	59.94	75393.28

Tabla C.125. Consumo del proceso total para el envío del paquete

C.3.1.3.3.- Envío de paquetes por broadcast y en modo no cifrado

Este ejemplo muestra como enviar paquetes con módulos DM.

El modo de envío se realizará por broadcast y las tramas no estarán cifradas. La dirección broadcast (0x000000000000FFFF) se especifica como dirección de destino.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 73 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto y el canal tiene que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se desactivará, poniéndolo a 0.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

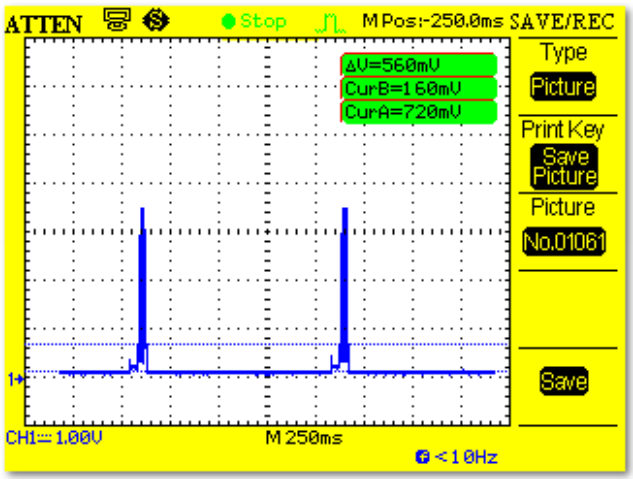


Fig. C.120.a. Medida general del proceso cada segundo

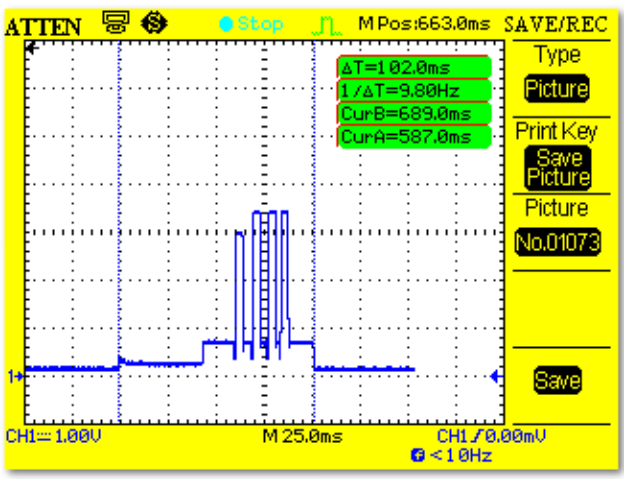


Fig. C.120.b. Tiempo de ejecución del proceso total

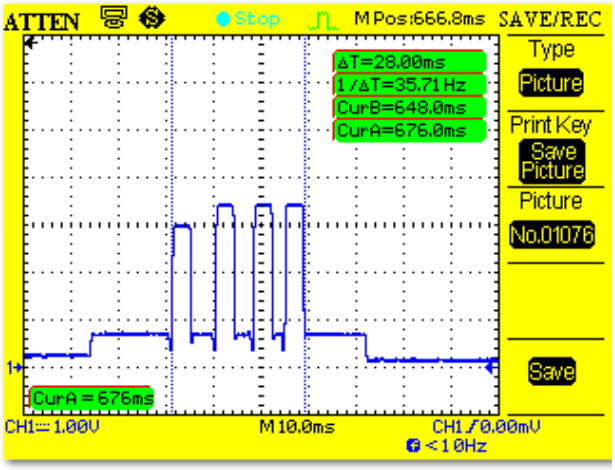


Fig. C.120.c. Reintentos en el envío del paquete

Como se observa en las gráficas (b) y (c), hay cuatro picos de envíos de datos. Esto es debido a que le comando MT, que define el número de transmisiones de difusión adicionales, por defecto, son tres. Como todos los paquetes de difusión se transmiten MT+1 veces para asegurarse de que se recibe, entonces se producen cuatro transmisiones, que son los cuatr picos que se ven ahí.

Este tipo de modo de transmisión no tiene ACK por lo tanto el emisor no recibirá ningún mensaje de que la transmisión del paquete se ha realizado correctamente, por lo tanto, no hay seguridad de que el paquete les haya llegado a todos los nodos de la red.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	102	77.86	7942.24

Tabla C.126. Consumo del proceso total para el envío del paquete

C.3.1.3.4.- Envío de paquetes por unicast y en modo cifrado

Este ejemplo muestra como enviar paquetes a un Gateway indicando la dirección MAC del módulo XBee receptor. El modo de envío se realizará por unicast y las tramas estarán cifradas. La dirección MAC del receptor es 0013A200406C16CE.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 73 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto y el canal tiene que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se activará, poniéndolo a 1.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

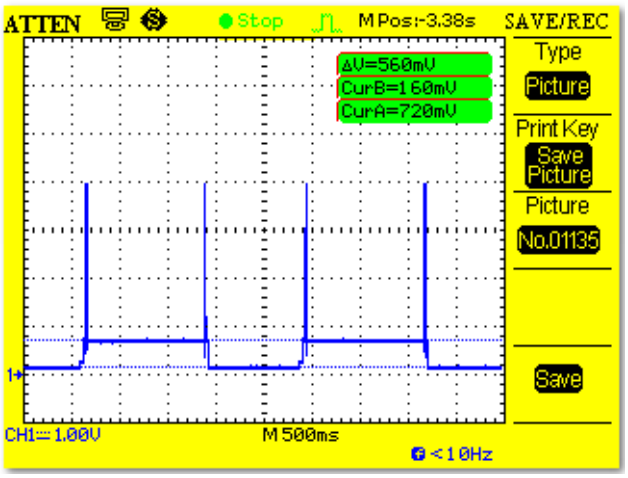


Fig. C.121.a. Medida general del proceso cada segundo

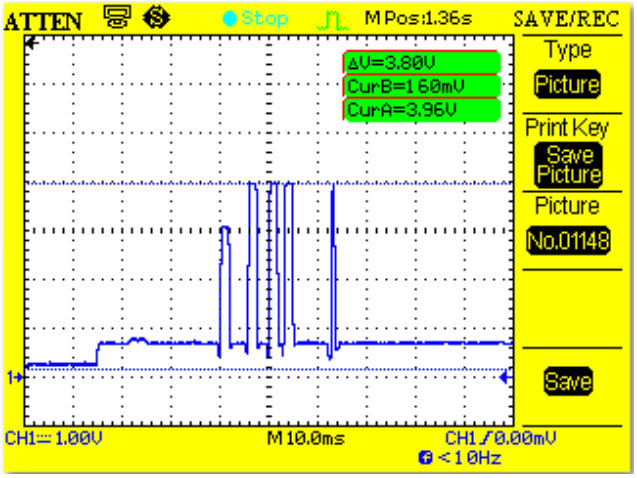


Fig. C.121.b. Consumo de los picos producidos al principio de la transmisión

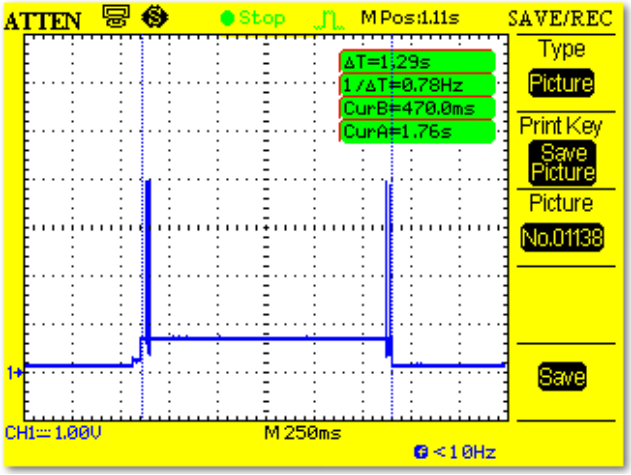


Fig. C.121.c. Proceso de transmisión y envío del paquete

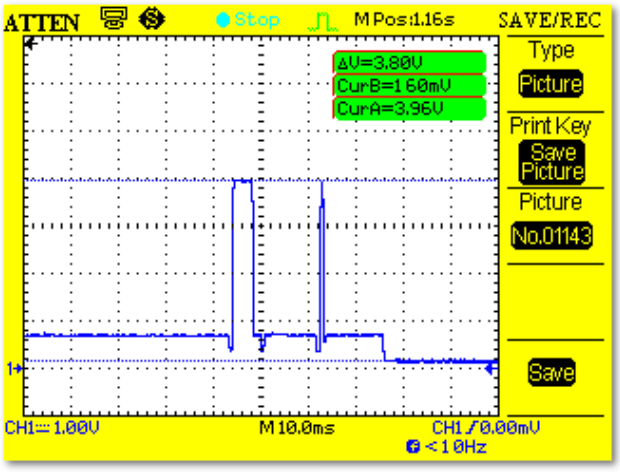


Fig. C.121d. Consumo envío del paquete

Como se ve en la gráfica (c), el tiempo del proceso de transmisión del paquete es ligeramente mayor con respecto al de sin cifrar debido a que, esta vez, el envío de las tramas se realiza en modo cifrado.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	1340	57.47	77022.4

Tabla C.127. Consumo del proceso total para el envío del paquete

C.3.1.3.5.- Comandos AT

Este ejemplo muestra como configurar los parámetros XBee básicos para comunicarse entre diferentes dispositivos XBee usando la misma red.

Para leer los comandos AT de configuración del XBee en necesario encender el módulo previamente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

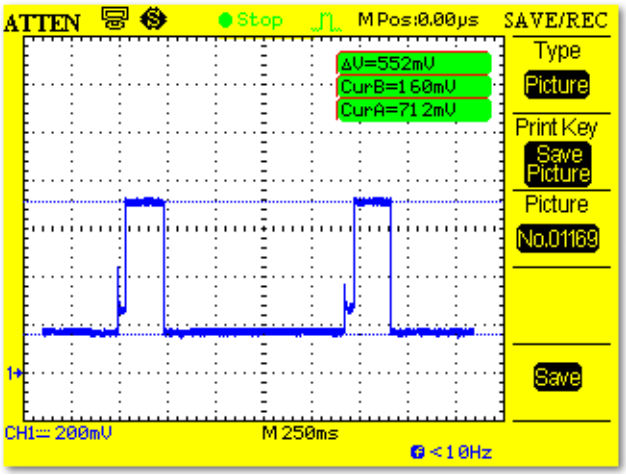


Fig. C.122.a. Medida general del proceso cada segundo

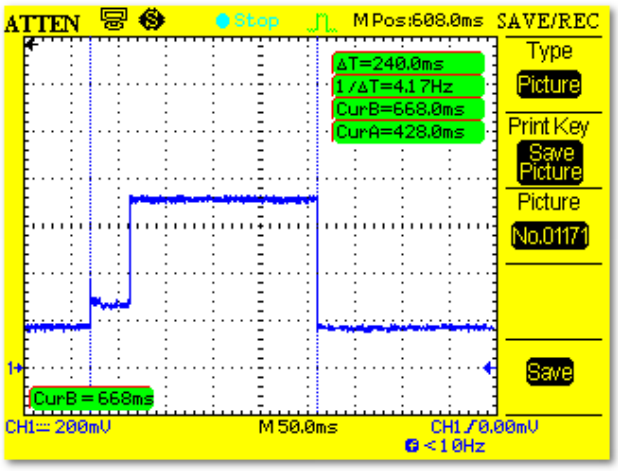


Fig. C.122.b. Tiempo de ejecución del proceso total

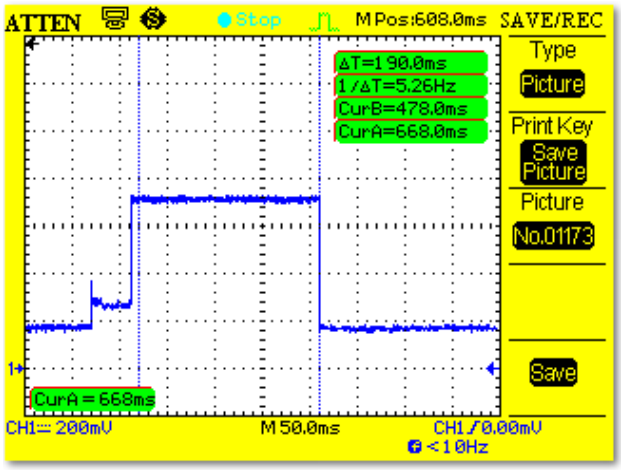


Fig. C.122.c. Tiempo de ejecución de los parámetros

Como se ve en la gráfica (c), el tiempo de ejecución de los comandos AT es de 190 ms, aunque no consumen nada cuando se enciende el módulo.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	240	47.6	11426.24

Tabla C.128. Consumo del proceso total de la configuración de los comandos AT

C.3.1.3.6. - Obtención del RSSI

Este ejemplo muestra cómo conseguir el valor del RSSI del último paquete recibido. En este protocolo, es posible extraer esta información de cada paquete. Al lado, hay una función API que muestra esta información. Antes de correr este ejemplo, hay que asegurarse de que hay otro emisor enviando paquetes al módulo XBee para recibir información.

Para que el valor del RSSI llegue correctamente, es necesario dejar encendido el XBee.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

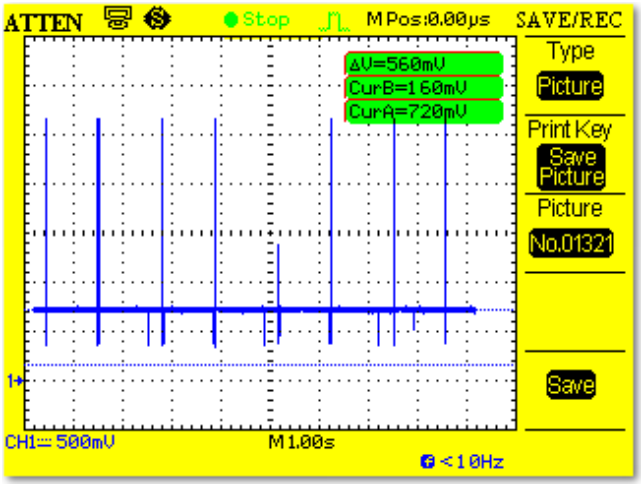


Fig. C.123.a. Medida general del proceso cada tres segundos

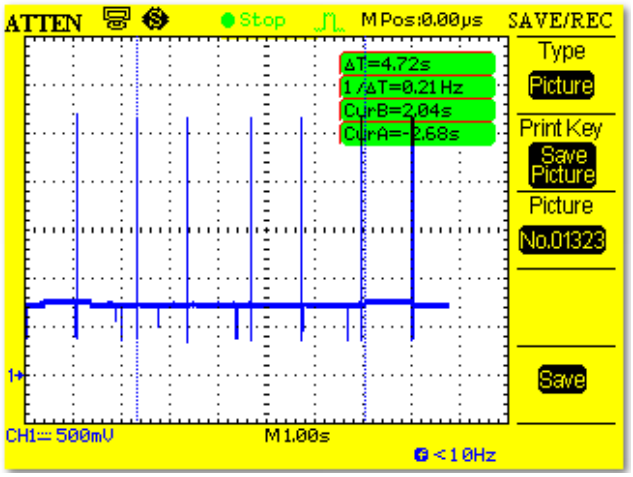


Fig. C.123.b. Tiempo de ejecución del proceso

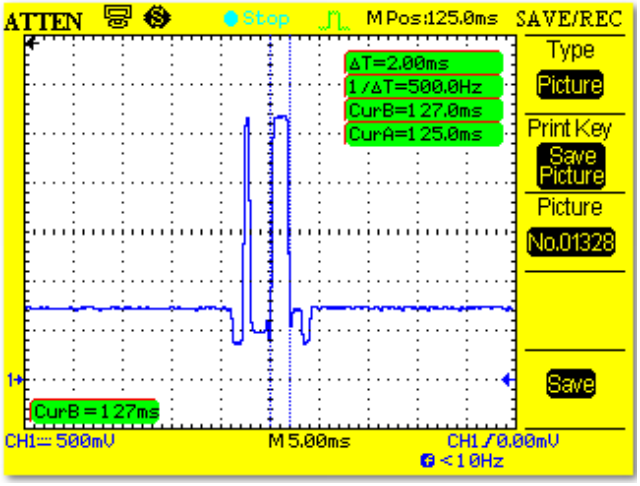


Fig. C.123.c. Tiempo de ejecución de cada pulso

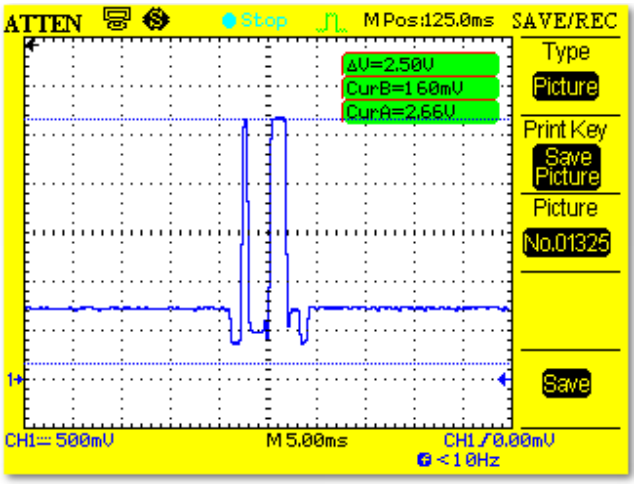


Fig. C.123.d. Consumo de los pulsos

Como se ve en la gráfica (a), la señal sale totalmente plana dificultando saber cuánto es el tiempo de ejecución del proceso. Para ello, se añadió una serie de funciones al código que produce un consumo extra a lo ejecutado y, así, distinguir mejor el tiempo de ejecución de este proceso, como se ve claramente en la gráfica (b).

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	4720	0.693	3272.64

Tabla C.129. Consumo del proceso de obtención del RSSI

C.3.1.4.- Módulo ZigBee

Para que se produzca la comunicación entre el coordinador y el router y que estén en la misma red, es necesario, previamente, configurar los módulos. Tienen que tener el mismo PAN ID y en el mismo canal de comunicación.

Primeramente, se configurará el coordinador. Previamente, se pone el PAN ID a cero para que el coordinador escanee que PAN ID está libre para conectarse y así, poder definir uno tanto en el coordinador como en el router. Te calcula el PAN ID de 16 bits y el canal al que se debe conectar.

Cabe destacar, que en este tipo de módulos, cuando se enchufa el XBee, se deja pasar tres segundos para comprobar cuando hay sincronización con el coordinador. Desde el estado de off, es raro que el ZB, se sincronice en menos de tres segundos.

C.3.1.4.1.- Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo ZigBee cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

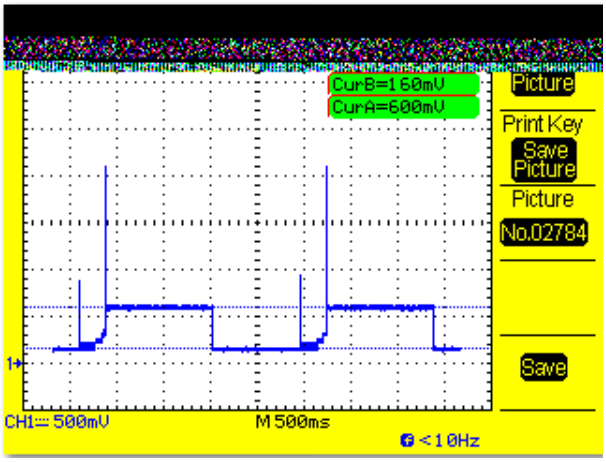


Fig. C.124. a. Medida general del proceso cada segundo

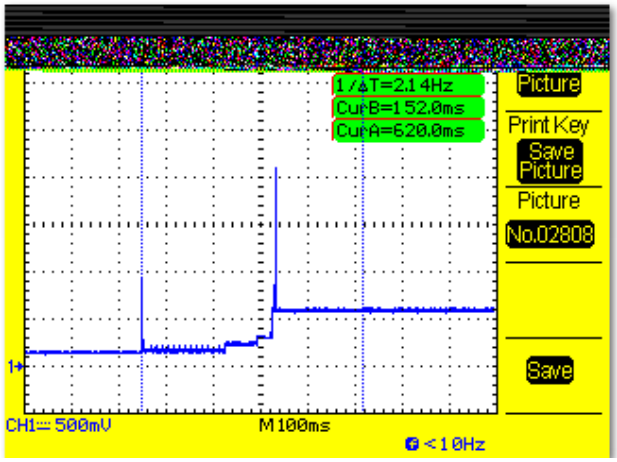


Fig. C.124.b. Proceso de encendido del módulo

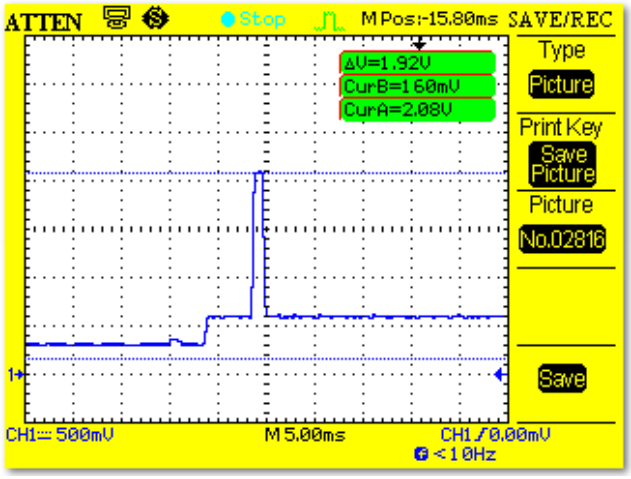


Fig. C.124.c. Pulso de corriente durante el encendido del módulo

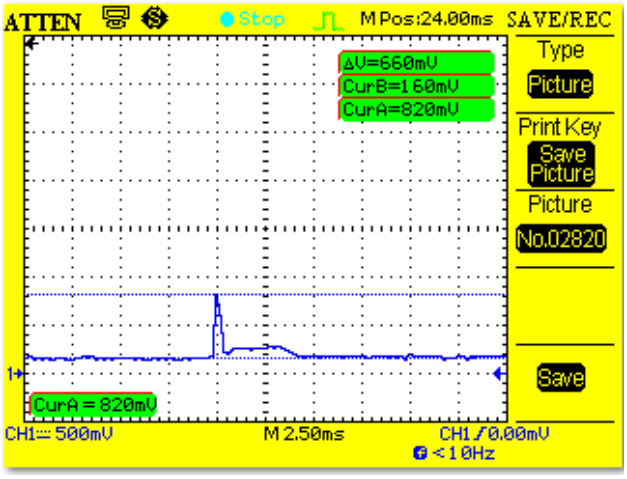


Fig. C.124.d. Zoom en el proceso de encendido

Como se ve en la gráfica (b), el tiempo de ejecución del proceso de encendido del módulo es de 470 ms, mientras que el proceso de apagado se realiza instantáneamente.

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	470	21.88	10284.52
Estado On	1000	44.4	44400
Proceso Off	0	44.4	0

Tabla C.130. Consumo del proceso y estado on y del proceso off del módulo ZigBee

C.3.1.4.2.- Conexión del Router con el Coordinador

Este ejemplo muestra como unir un dispositivo router a una red conocida. Una vez sacado el PAN ID de 16 y 64 bits y el canal al cargar los parámetros en el coordinador, se introducirán en el router para que se produzca la conexión entre ellos y así, estarán en la misma red de comunicación.

El PAN ID de 64 bits es el 0xFCE05D05FEC97504.

El PAN ID de 16 bits es el 0x162F.

El canal es el 0x0E.

A continuación, se muestran la gráfica como resultado del ejemplo ejecutado:

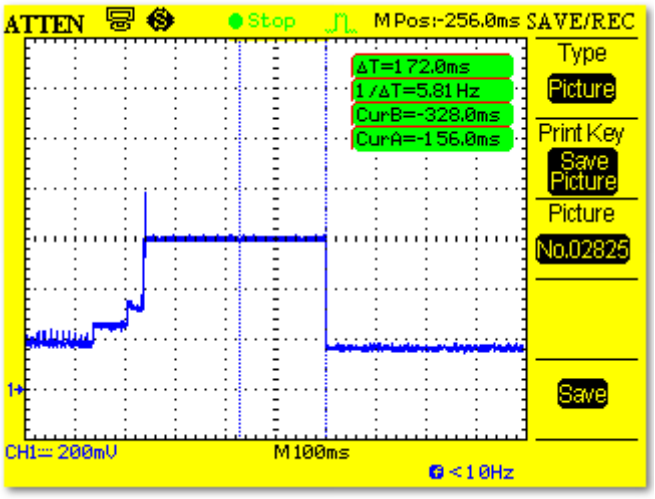


Fig. C.125. Tiempo de ejecución de la conexión entre el router y el coordinador

Como se ve en la gráfica, el tiempo de ejecución de la conexión entre el router y el coordinador es de 172 ms aunque no se produce un consumo extra después de encender el módulo.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	642	27.91	17921.32

Tabla C.131. Consumo del proceso de conexión con el coordinador

C.3.1.4.3.- Envío de paquetes por Unicast y en modo no cifrado

Este ejemplo muestra como enviar paquetes a un Gateway indicando la dirección MAC del módulo XBee receptor. El modo de envío se realizará por unicast y las tramas no estarán cifradas. La dirección MAC del receptor es 0013A200408C9DB6.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 74 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

Para poder realizar este tipo de envíos, primeramente se debe configurar una serie de parámetros para los dos módulos, tanto para el coordinador como para el router. Como se ha comentado anteriormente, los dos nodos tienen que estar en la misma red para que se establezca la comunicación. Por lo tanto, se configuraran con el mismo PAN ID y canal. Al enviarse el paquete sin cifrar, se desactivará el modo cifrado también, poniéndose a 0 en los dos módulos.

El PAN ID de 64 bits es el 0xFCE05D05FEC97504.

El PAN ID de 16 bits es el 0xD272.

El canal es el 0x11.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

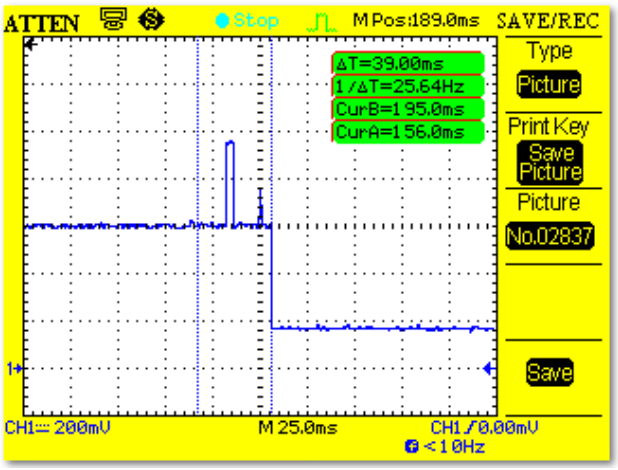


Fig. C.126.a. Proceso de transmisión y envío del paquete

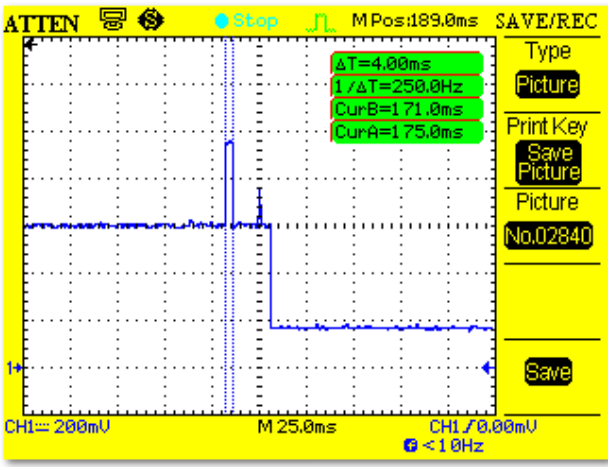


Fig. C.126.b. Tiempo del envío del paquete

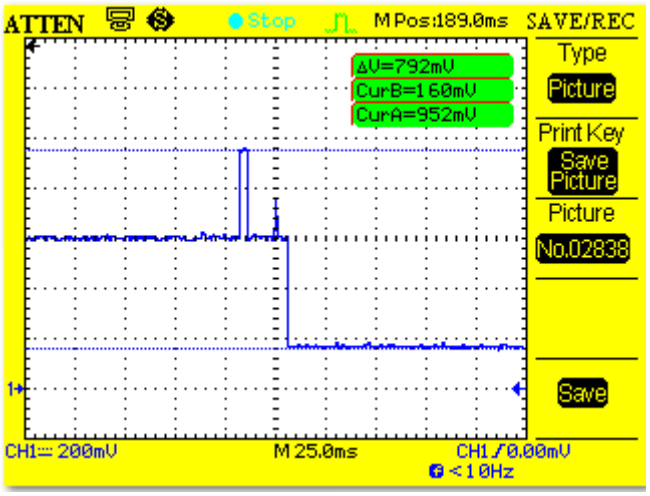


Fig. C.126.c. Consumo envío del paquete

Como se ve en la gráfica (a), durante el proceso de transmisión del paquete, hay un tiempo previo y posterior al envío del paquete.

El momento previo al envío es debido a que los módulos tienen que escuchar el canal al que se ha conectado y, cuando comprueban que no hay nadie transmitiendo, se lanzan a enviar.

El momento posterior al envío del paquete corresponde al ACK. Todos los XBees y usando este modo de transmisión lo tienen. Es un paquete que el receptor envía al emisor como confirmación de que aquel ha recibido correctamente el paquete enviado.

En las gráficas (b) y (c), corresponde al envío del paquete. Normalmente, sale un pico con una diferencia de consumo considerable aunque el tiempo es mínimo.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	3796	41.65	158131.72

Tabla C.132. Consumo del proceso total para el envío del paquete

C.3.1.4.4.- Envío de paquetes por Broadcast y en modo no cifrado

Este ejemplo muestra como enviar paquetes con módulos ZigBee.

El modo de envío se realizará por broadcast y las tramas no estarán cifradas. La dirección broadcast (0x000000000000FFFF) se especifica como dirección de destino.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 92 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto y el canal tienen que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se desactivará, poniéndolo a 0 para los dos módulos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

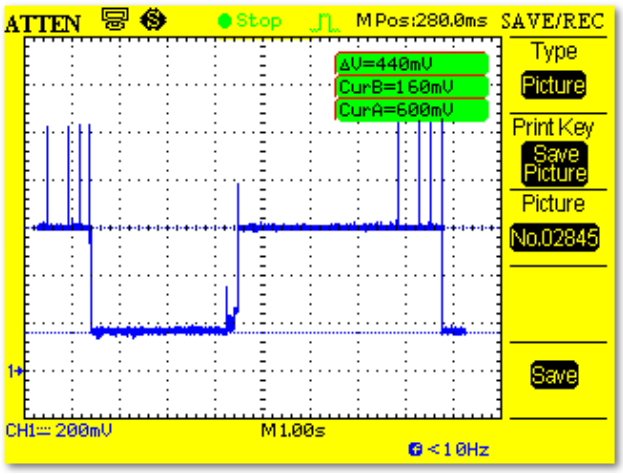


Fig. C.127.a. Medida general del proceso cada tres segundos

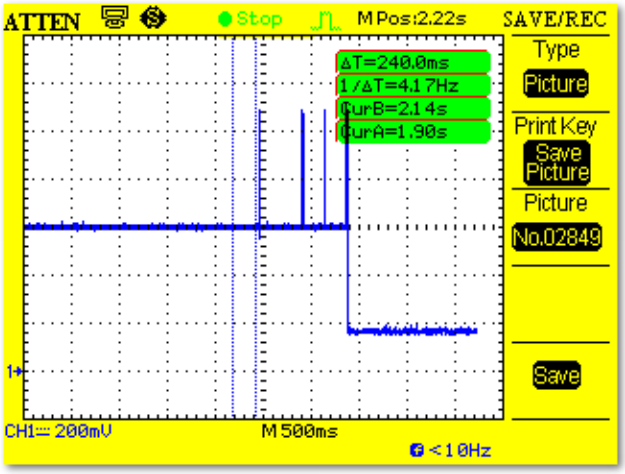


Fig. C.127.b. Tiempo de ejecución de la comprobación de los parámetros de red

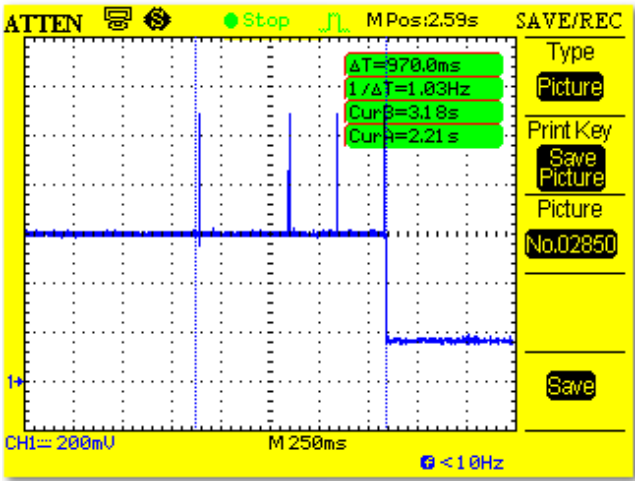


Fig. C.127.c. Proceso de transmisión y envío del Paquete

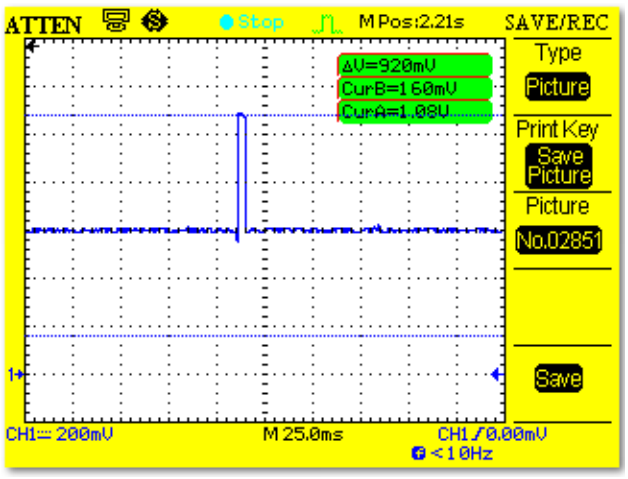


Fig. C.127.d. Consumo pico de inicio previo a la retransmisión

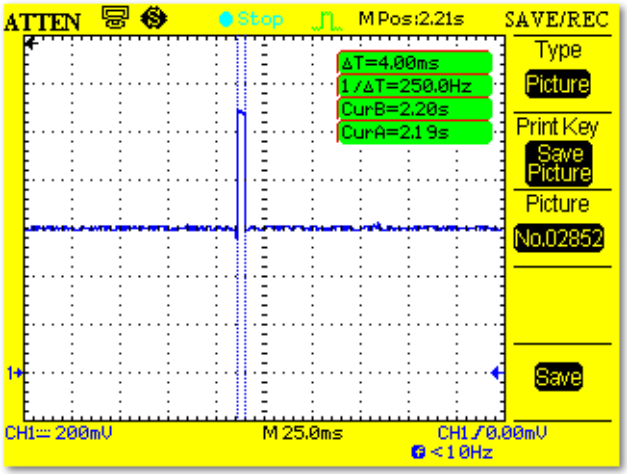


Fig. C.127.e. Tiempo del pulso de inicio previo a la Retransmisión

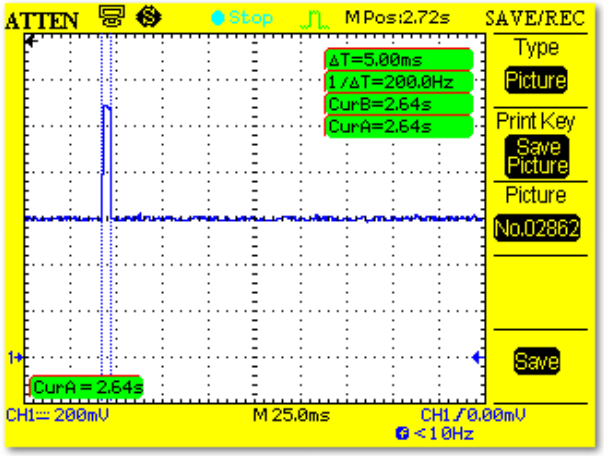


Fig. C.127.f. Tiempo del pulso de una de las retransmisiones

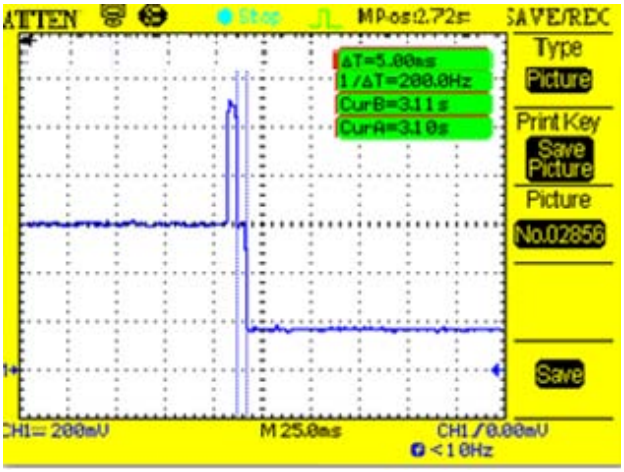


Fig. C.127.g. Tiempo de post-procesado

Este modo de envío retransmite el paquete tres veces que, como se observa en la gráfica (c),

son los tres pulsos que se ven después del pulso de inicio previo a las retransmisiones.

Este tipo de modo de transmisión no tiene ACK por lo tanto el emisor no recibirá ningún mensaje de que la transmisión del paquete se ha realizado correctamente, por lo tanto, no hay seguridad de que el paquete les haya llegado a todos los nodos de la red y por eso mismo, como se ve en la gráfica (g), el tiempo del post-procesado es mínimo.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	4680	42.42	198558.9

Tabla C.133. Consumo del proceso total para el envío del paquete

C.3.1.4.5.- Envío de paquetes por Unicast y en modo cifrado

Este ejemplo muestra como enviar paquetes a un Gateway indicando la dirección MAC del módulo XBee receptor. El modo de envío se realizará por unicast y las tramas estarán cifradas. La dirección MAC del receptor es 0013A200408C9DB6.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 66 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto y el canal tienen que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado se activará, poniéndolo a 1 en los dos módulos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado, enviándose el paquete **con una transmisión**:

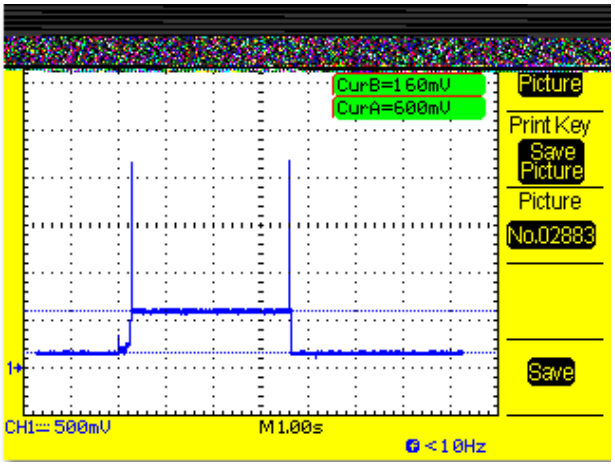


Fig. C.128.a. Medida general del proceso

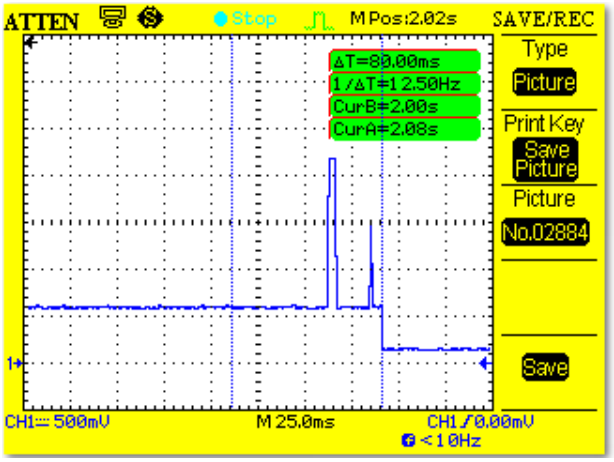


Fig. C.128.b. Proceso de transmisión y envío del Paquete

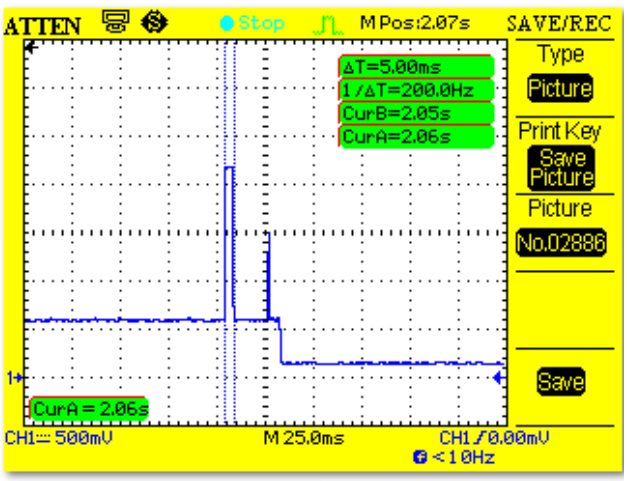


Fig. C.128.c. Tiempo de envío del paquete

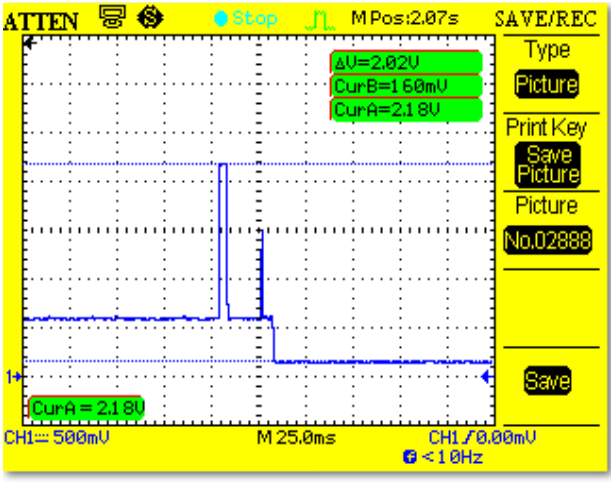


Fig. C.128.d. Consumo envío del paquete

Como se ve en la gráfica (b), el tiempo del proceso de transmisión del paquete en este modo es mayor con respecto al anterior sin cifrar debido a que, esta vez, el envío de las tramas se realiza en modo cifrado.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	3790	41.83	158572.52

Tabla C.134. Consumo del proceso total para el envío del paquete

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado, necesitándose **dos retransmisiones** para el envío de los datos:

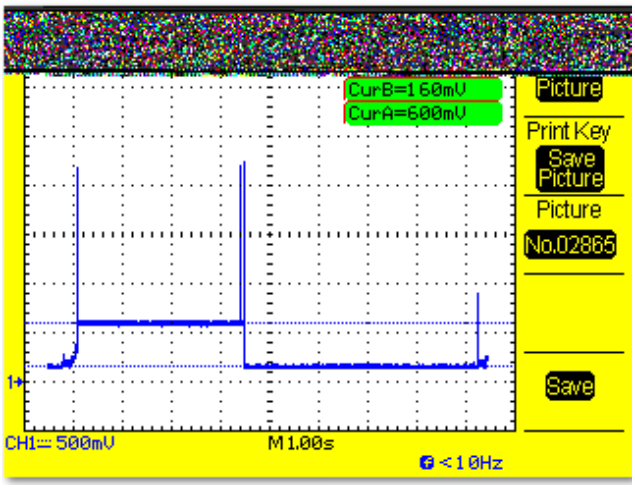


Fig. C.129.a. Medida general del proceso

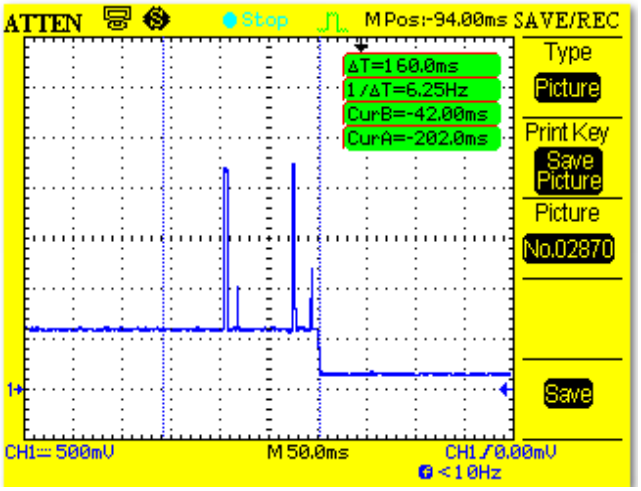


Fig. C.129.b. Proceso de transmisión y envío del paquete

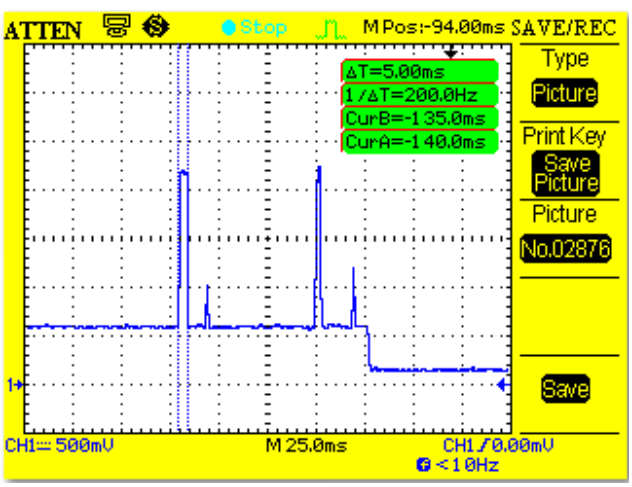


Fig. C.129.c. Tiempo del pulso de una de las retransmisiones

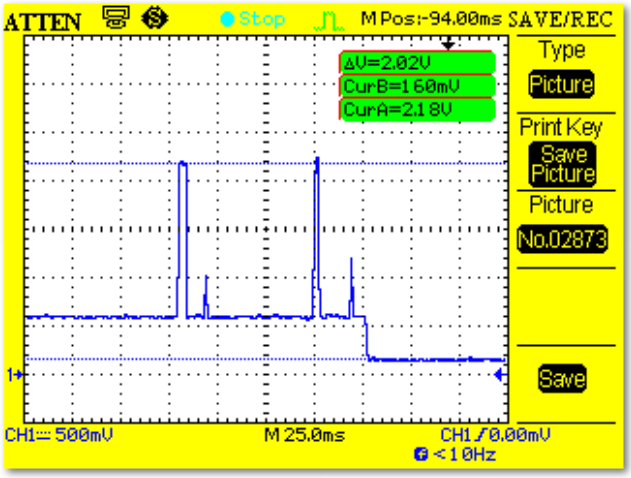


Fig. C.129.d. Consumo de los pulsos de las retransmisiones

Las dos retransmisiones que se producen es debido a que, si el módulo emisor no recibe un mensaje del ACK por parte del receptor de que el paquete de datos lo ha recibido correctamente, vuelve a retransmitir por segunda vez hasta que lo recibe.

Como se ve en la gráfica (b), el tiempo del proceso de transmisión, desde que se pone a enviar el paquete hasta quelo envía y se recibe el ACK, es el doble con respecto al anterior, como es lógico, ya que retransmite dos veces.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	3870	42.09	162906.52

Tabla C.135. Consumo del proceso total para el envío del paquete

C.3.1.4.6.- Envío de paquetes por Broadcast y en modo cifrado

Este ejemplo muestra como enviar paquetes con módulos ZigBee.

El modo de envío se realizará por broadcast y las tramas estarán cifradas. La dirección broadcast (0x000000000000FFFF) se especifica como dirección de destino.

Con respecto a la carga útil de datos, se ha optado por poner el máximo posible para este método, es decir, 84 bytes, añadiendo los bytes necesarios en las tramas para tal fin.

El PAN ID impuesto y el canal tienen que ser el mismo que el anterior, para establecer la comunicación. El modo cifrado desactivará, poniéndolo a 1 en los dos módulos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

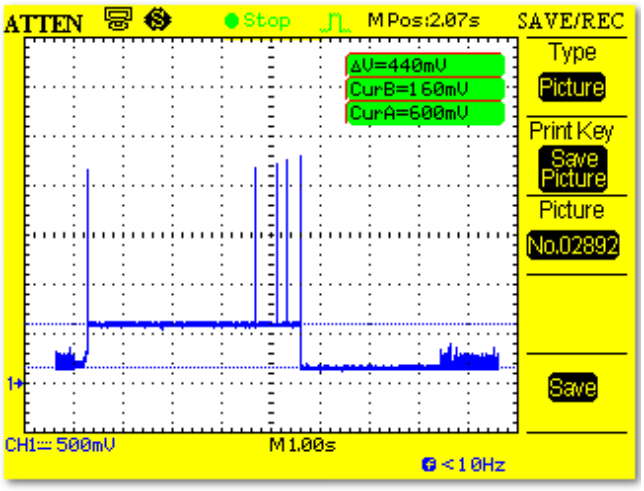


Fig. C.130.a. Medida general del proceso cada tres segundos

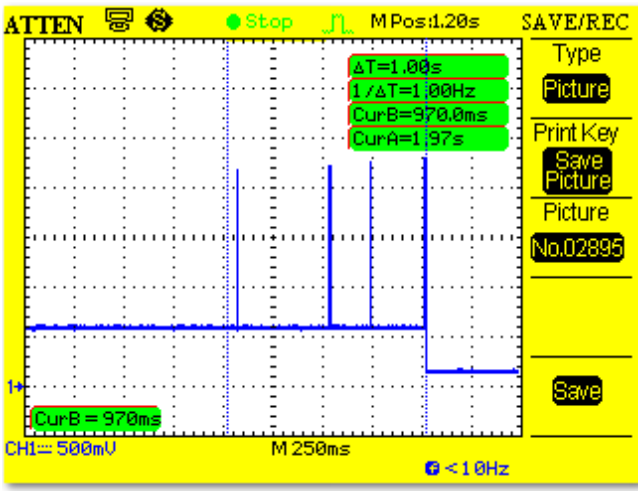


Fig. C.130.b. Proceso de transmisión y envío del paquete

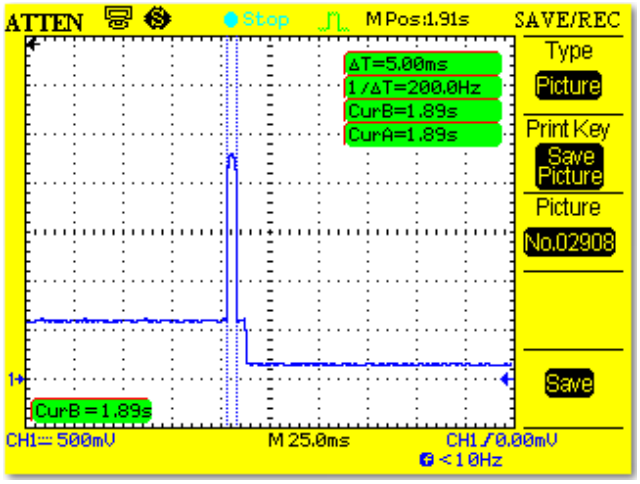


Fig. C.130.c. Tiempo del pulso de una de las retransmisiones

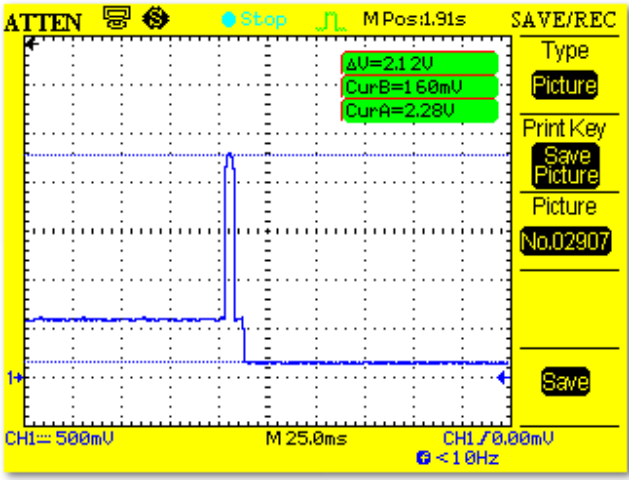


Fig. C.130.d. Consumo del pulso de una de las retransmisiones

Como se ve en la gráfica (b), el tiempo del proceso de transmisión del paquete es ligeramente mayor con respecto al anterior sin cifrar debido a que, esta vez, el envío de las tramas se realiza en modo cifrado.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	4710	42.7	201134.52

Tabla C.136. Consumo del proceso total para el envío del paquete

C.3.1.4.7.- Configuración del modo cifrado en el router

Este ejemplo muestra cómo configurar el modo cifrado en el módulo con el firmware del router.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

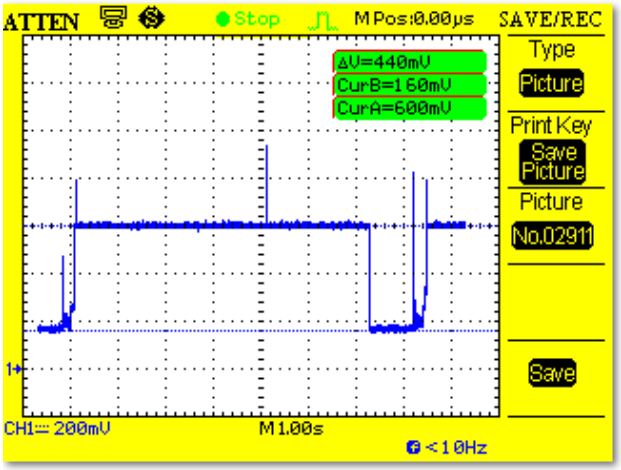


Fig. C.131.a. Medida general del proceso cada segundo

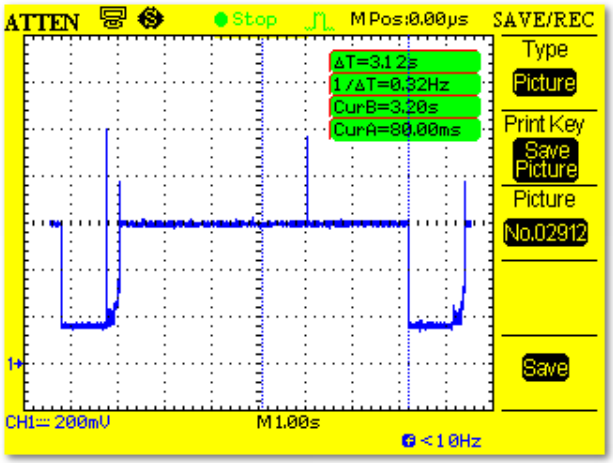


Fig. C.131.b. Tiempo de ejecución de los parámetros de configuración del cifrado

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	6590	42.8	282086.12

Tabla C.137. Consumo del proceso total de la configuración en modo cifrado en el router

C.3.1.4.8.- Configuración del modo cifrado en el coordinador

Este ejemplo muestra cómo configurar el modo cifrado en el módulo coordinador.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

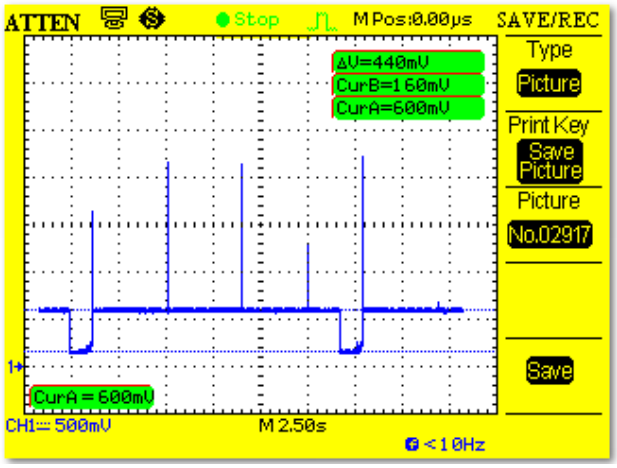


Fig. C.132.a. Medida general del proceso cada segundo

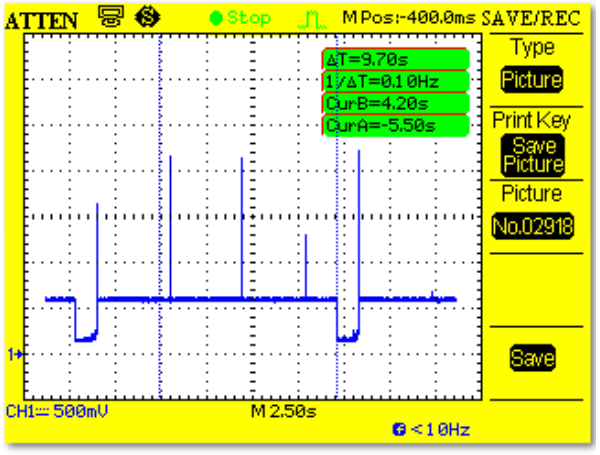


Fig. C.132.b. Tiempo de ejecución del proceso de configuración del cifrado

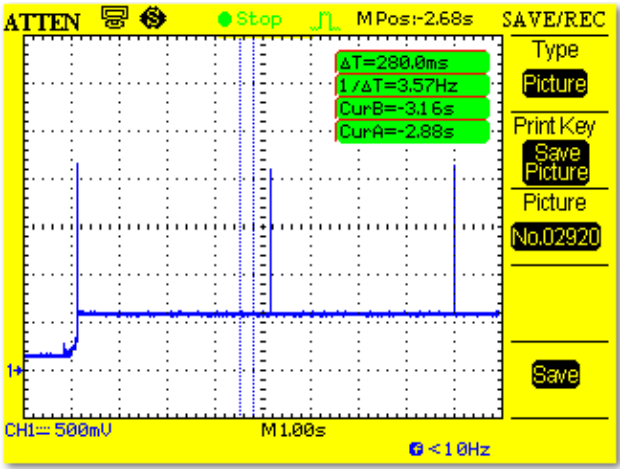


Fig. C.132.c. Tiempo de ejecución de los parámetros de configuración del cifrado

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	13990	43.7	611496.52

Tabla C.138. Consumo del proceso total para la configuración del cifrado en el coordinador

C.3.2. Módulo Bluetooth

Antes de empezar a usar el módulo, necesita ser inicializado. Durante este proceso, serán enviados los parámetros de configuración al módulo y también se inicializará la tarjeta SD, con lo cual habrá que ponerla en el Waspmote desde el principio.

Para este tipo de módulo, se dejará encendido el módulo en el setup () y se calculará el consumo extra que produce cada acción desarrollada.

C.3.2.1. – Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo Bluetooth PRO cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

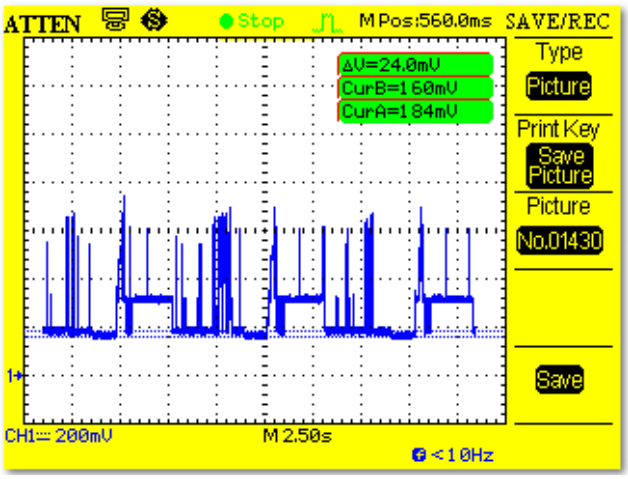


Fig. C.133.a. Medida general del proceso cada segundo

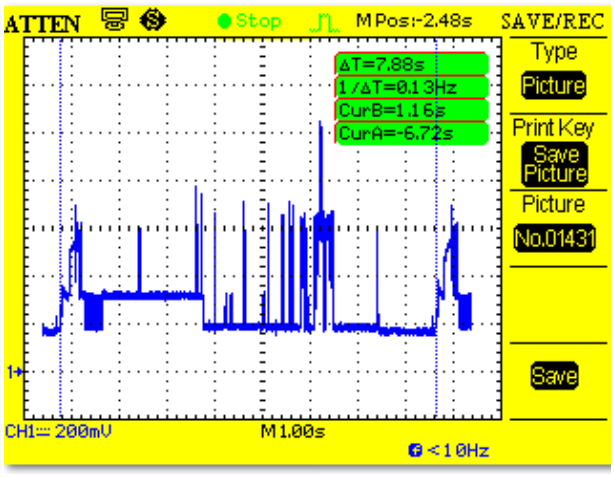


Fig. C.133.b. Tiempo de ejecución del proceso completo

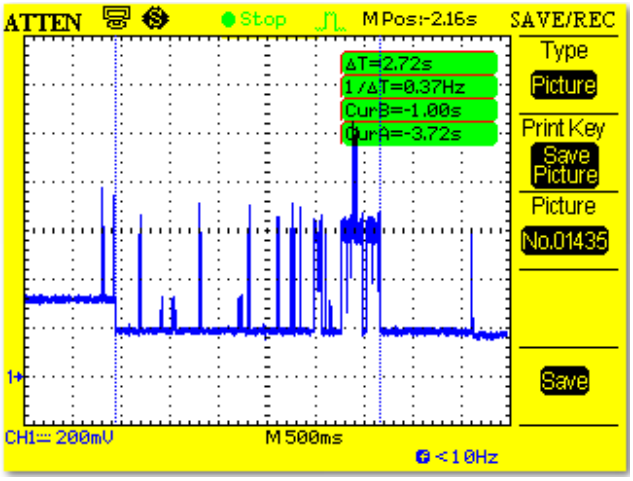


Fig. C.133.c. Proceso de inicialización del módulo

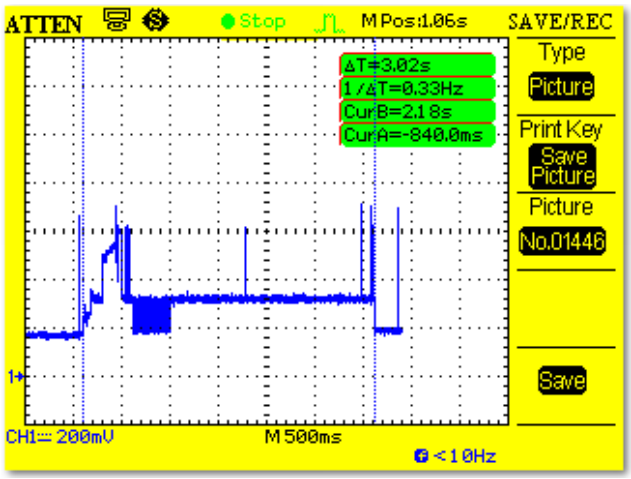


Fig. C.133.d. Proceso de reseteo del módulo

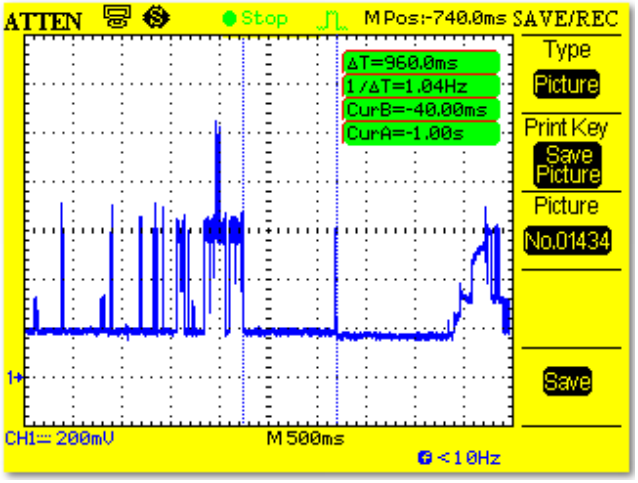


Fig. C.133.e. Estado on del módulo

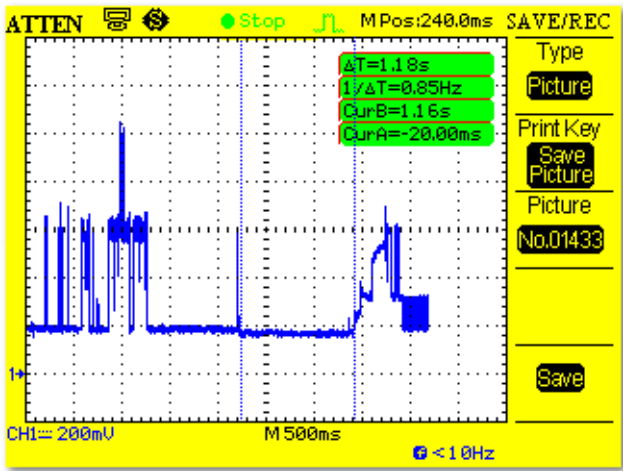


Fig. C.133.f. Proceso off y estado off del módulo

Como se ve en las imágenes, el proceso a on del módulo para por diferentes fases. Primero (d) se resetea el módulo para borrar lo que había anteriormente y evitar errores. Segundo (c y d), se prepara a la tarjeta SD y seguidamente se inicializa el módulo. La gráfica (e) y (f) corresponde con el estado on del módulo y el proceso off y estado off del módulo, respectivamente.

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	5800	15.03	87221.6
Estado On	1000	2.8	2800
Proceso Off	200	2	400

Tabla C.139. Consumo del proceso y estado on y del proceso off del módulo

C.3.2.2. – Escaneo normal

Este ejemplo muestra cómo hacer un escaneo normal con el módulo Bluetooth imprimiendo el número de dispositivos descubiertos indicando sus direcciones MAC y guardándolos en la tarjeta SD.

En este ejemplo, se realizará un escaneo de cinco segundos y a una potencia máxima.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

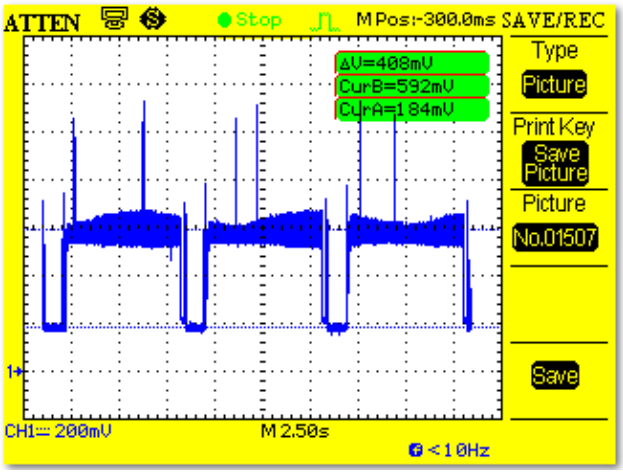


Fig. C.134.a. Medida general del proceso cada segundo

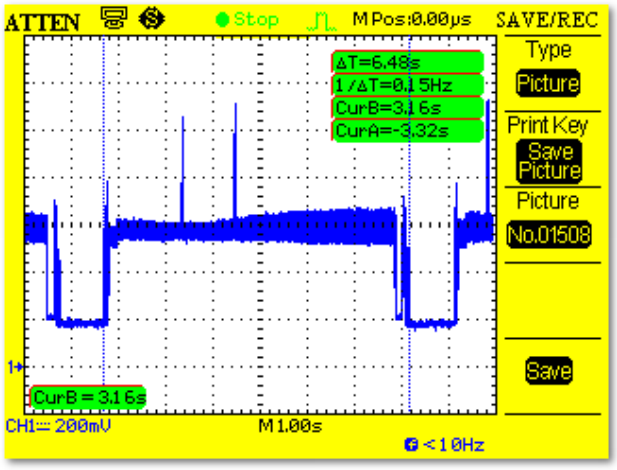


Fig. C.134.b. Tiempo de ejecución del proceso completo

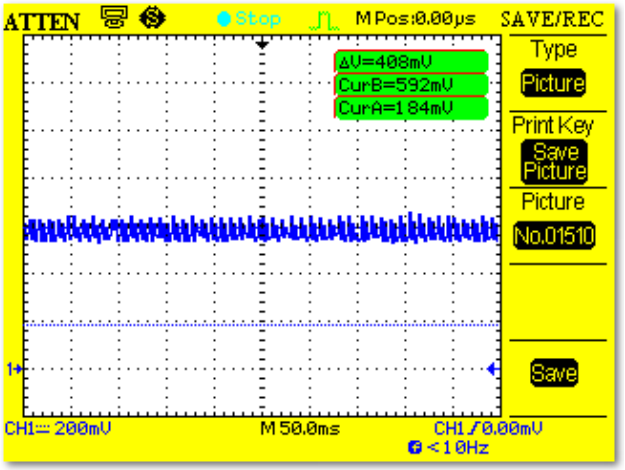


Fig. C.134.c. Consumo durante el escaneo

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	6480	40.8	264384

Tabla C.140. Consumo del proceso total del escaneo de dispositivos Bluetooth

C.3.2.3. – Escaneo limitado

Este ejemplo muestra cómo hacer un escaneo limitado con el módulo Bluetooth a máxima potencia, imprimiendo el número de dispositivos descubiertos con sus direcciones MAC y guardándolos en la tarjeta SD.

Se puso como límite para que se descubrieran dos dispositivos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

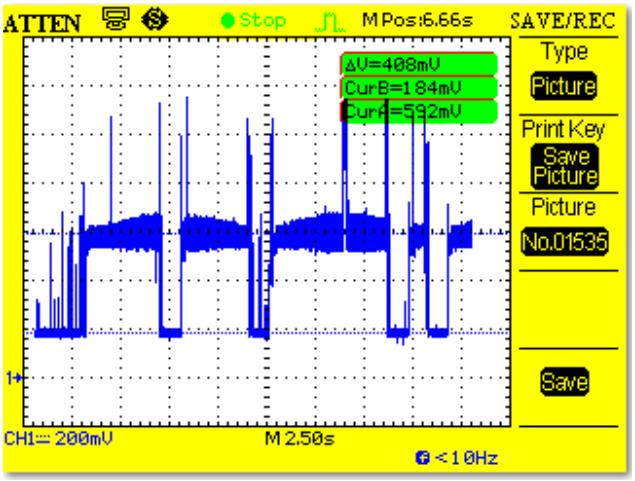


Fig. C.135.a. Medida general del proceso cada segundo

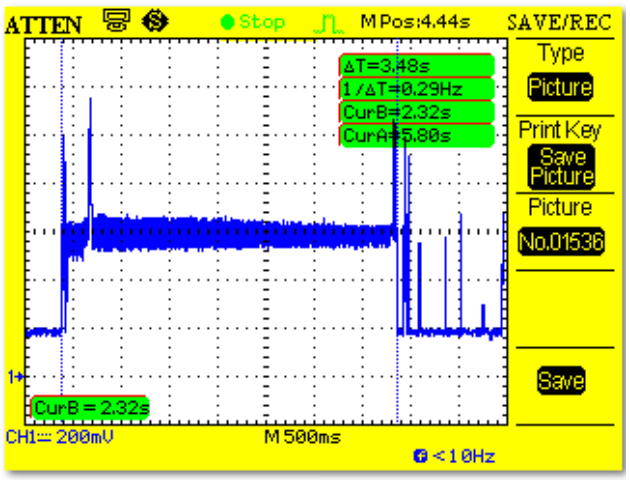


Fig. C.135.b. Tiempo de ejecución del proceso completo

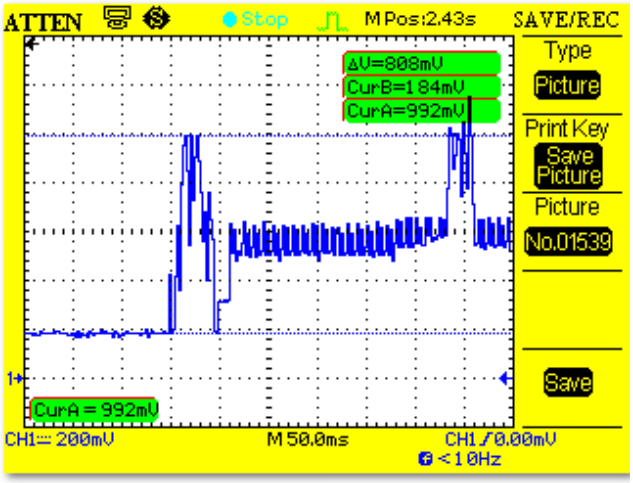


Fig. C.135.c. Inicio de proceso de escaneo

Como se ve en la gráfica (a), el tiempo de escaneo de los dispositivos puede variar. En algunas ocasiones les cuesta más y en otras menos.

Para el cálculo del consumo, se ha cogido el peor caso (b), es decir, cuando el proceso de escaneo es mayor que en otros.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	3480	40.8	141984

Tabla C.141. Consumo del proceso total del escaneo de dispositivos Bluetooth

C.3.2.4. – Escaneo con nombre descriptivo

Este ejemplo muestra como escanear dispositivos Bluetooth con el nombre de cada dispositivo a máxima potencia.

El tiempo de escaneo de los dispositivos es de cinco segundos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

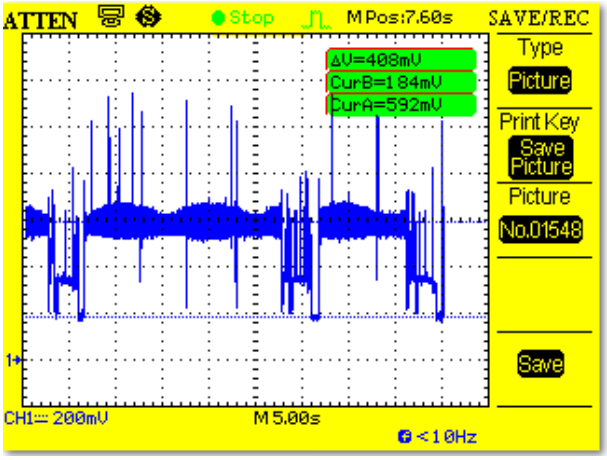


Fig. C.136.a. Medida general del proceso cada segundo

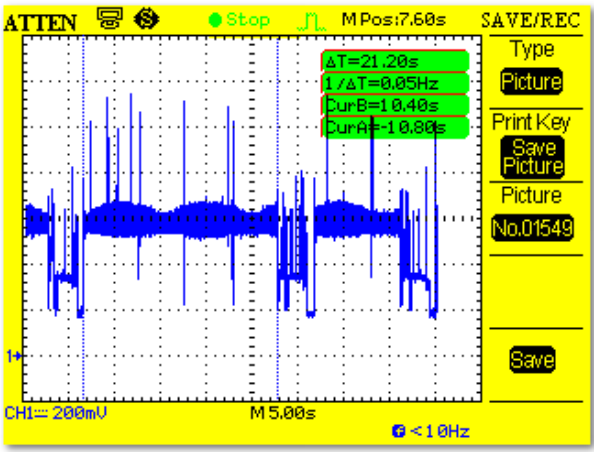


Fig. C.136.b. Tiempo de ejecución del escaneo

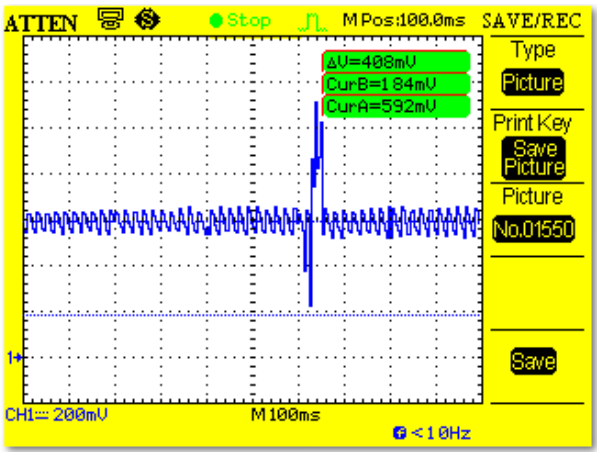


Fig. C.136.c. Consumo durante el escaneo

Como se ve en la gráfica (a), el tiempo de escaneo de los dispositivos puede variar. En algunas ocasiones les cuesta más y en otras menos.

Para el cálculo del consumo, se ha cogido el peor caso (b), es decir, cuando el proceso de escaneo es mayor que en otros.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	24200	37.67	911760

Tabla C.142. Consumo del proceso total del escaneo de dispositivos Bluetooth

C.3.2.5. – Escaneo de un dispositivo específico

Este ejemplo muestra como mirar un dispositivo específico usando su dirección MAC a la máxima potencia.

Se puso como dirección MAC mi dispositivo Bluetooth.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

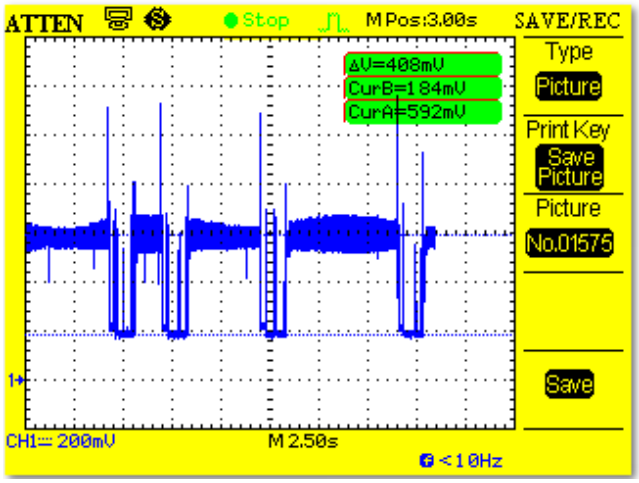


Fig. C.137.a. Medida general del proceso cada segundo

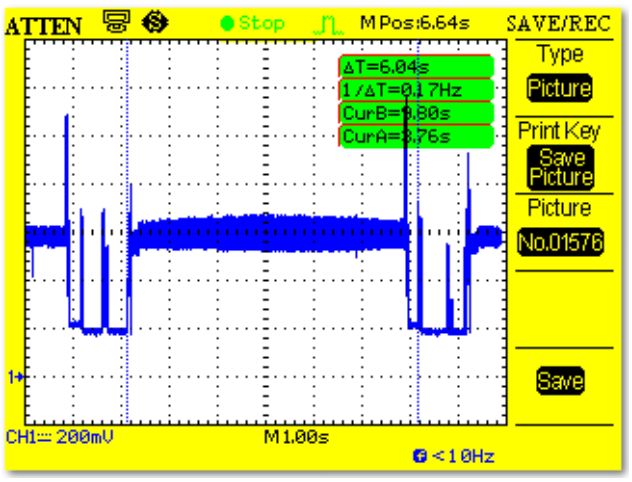


Fig. C.137.b. Tiempo de ejecución del proceso

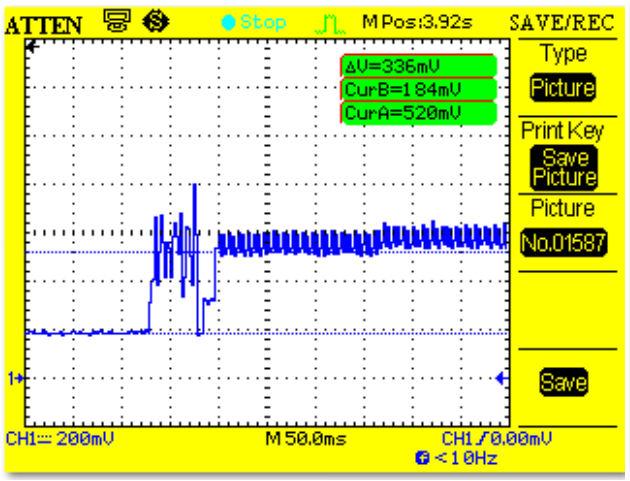


Fig. C.137.c. Inicio de proceso de escaneo

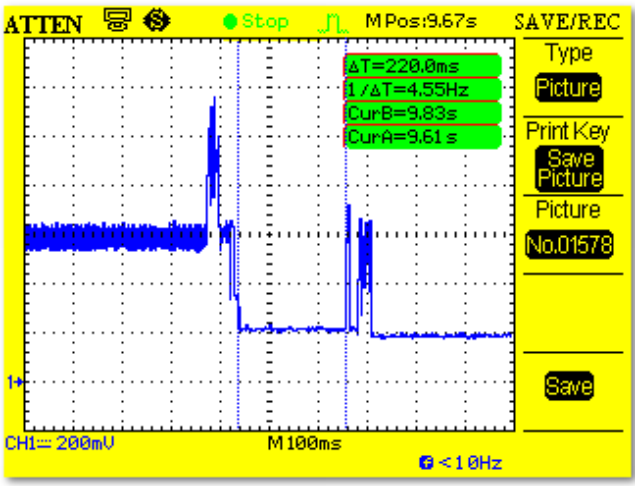


Fig. C.137.d. Final del proceso de escaneo

Como se ve en la gráfica (a), el tiempo de escaneo de los dispositivos puede variar. En algunas ocasiones les cuesta más y en otras menos.

Para el cálculo del consumo, se ha cogido el peor caso (b), es decir, cuando el proceso de escaneo es mayor que en otros.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	6040	39.15	236468

Tabla C.143. Consumo del proceso total del escaneo de dispositivos Bluetooth

C.3.2.6. – Creación de una conexión transparente

Este ejemplo crea un enlace transparente con un módulo Bluetooth remoto. El escaneo del dispositivo se realiza a máxima potencia.

Se especifica la dirección MAC del módulo receptor.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

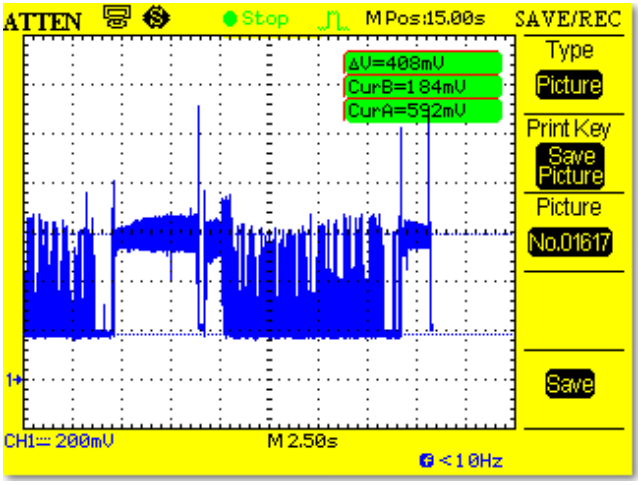


Fig. C.138.a. Medida general del proceso cada segundo

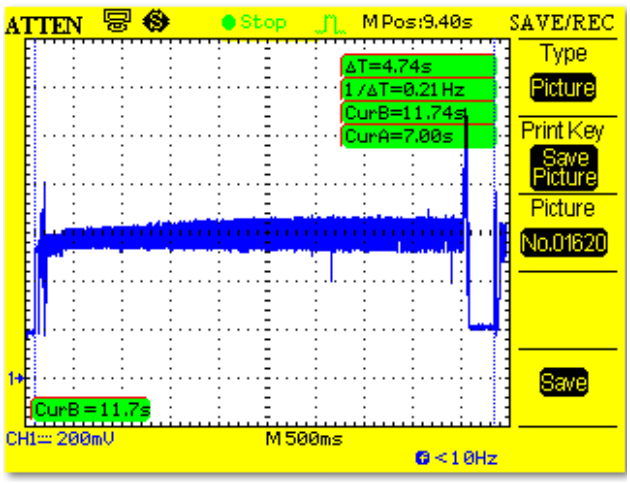


Fig. C.138.b. Tiempo de ejecución del escaneo

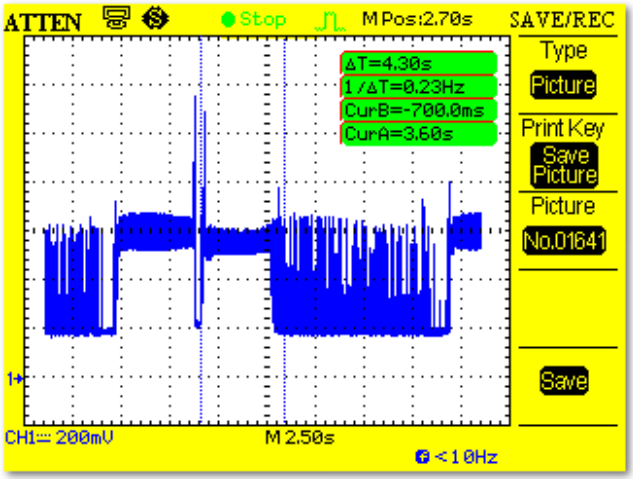


Fig. C.138.c. Tiempo de ejecución del Dispositivo

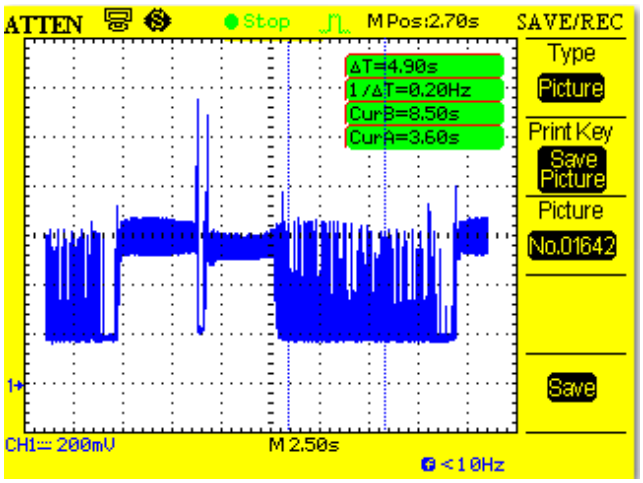


Fig. C.138.c. Tiempo de conexión con el dispositivo encontrado

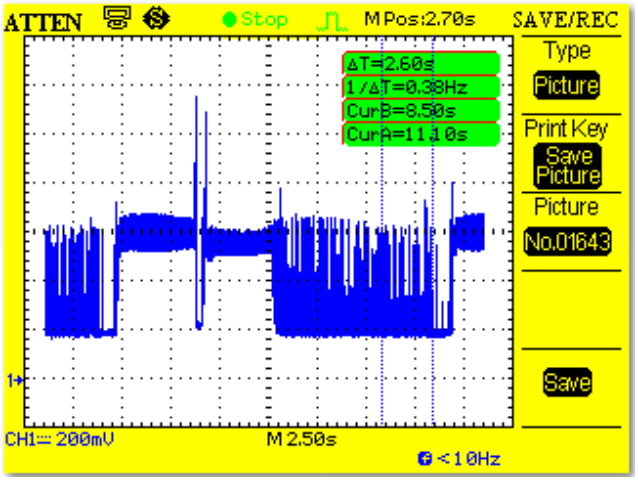


Fig. C.138.e. Tiempo de obtención del RSSI

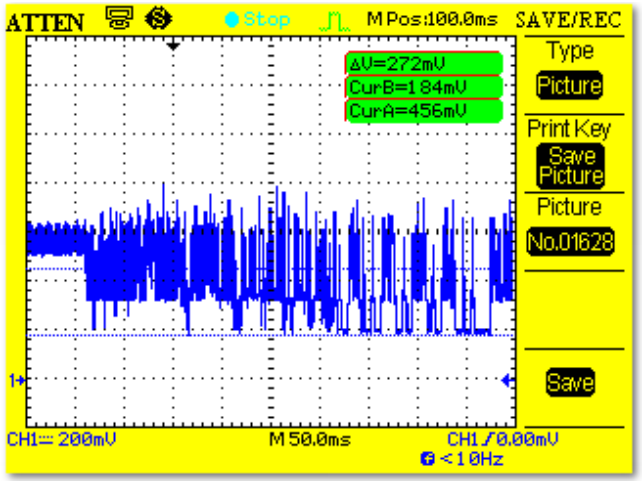


Fig. C.138.f. Proceso de conexión con el dispositivo

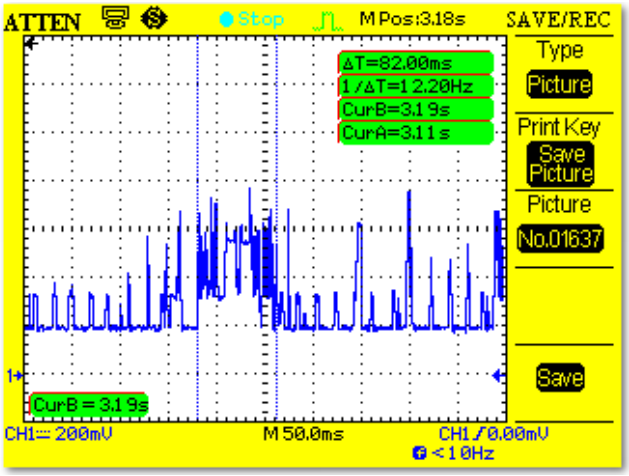


Fig. C.138.g. Proceso de obtención del RSSI

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	16540	33.32	551232

Tabla C.144. Consumo del proceso total de creación de una conexión transparente

C.3.3. Módulo BLE (Bluetooth Low Energy)

Para este módulo no es necesario conectar la tarjeta SD. También dejamos para las pruebas el módulo BLE encendido en el setup ().

C.3.3.1. – Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo BLE cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

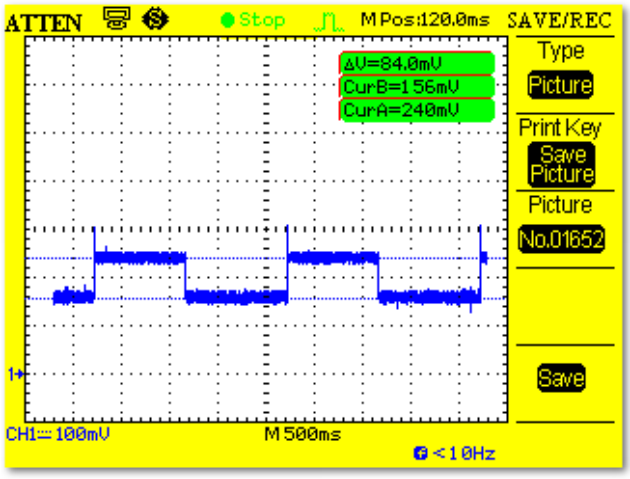


Fig. C.139.a. Medida general del proceso

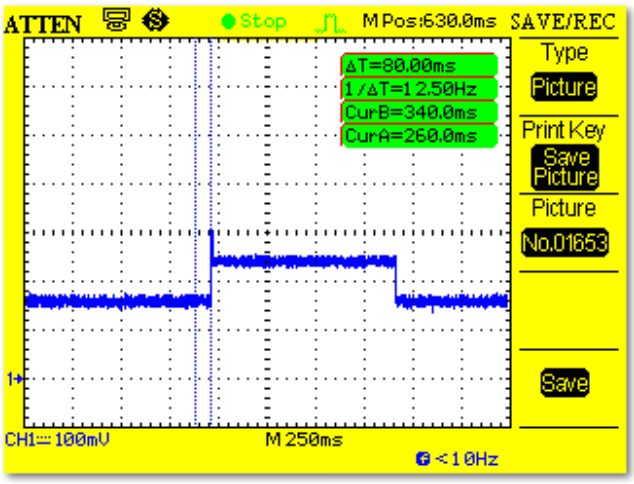


Fig. C.139.b. Proceso de encendido del módulo

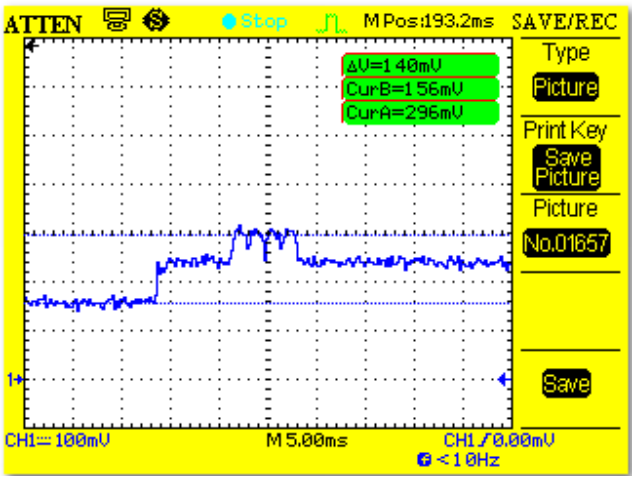


Fig. C.139.c. Pico de encendido del módulo

Como se ve en la gráfica (b), el tiempo de ejecución del proceso de encendido del módulo es de 80 ms, mientras que el proceso de apagado se realiza instantáneamente.

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	80	2.33	186.4
Estado On	1000	8.4	8400
Proceso Off	0	8.4	0

Tabla C.145. Consumo del proceso y estado on y del proceso off del módulo BLE

C.3.3.2. Escaneo normal

Este ejemplo muestra cómo hacer un escaneo normal con el módulo BLE imprimiendo el número de dispositivos descubiertos indicando sus direcciones MAC y guardando el resultado en la memoria EEPROM.

En este ejemplo, se realizará un escaneo de cinco segundos y a una potencia máxima.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

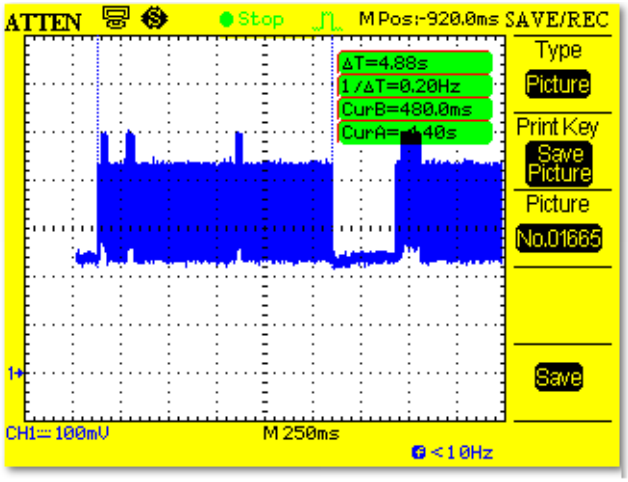


Fig. C.140.a. Medida general del proceso

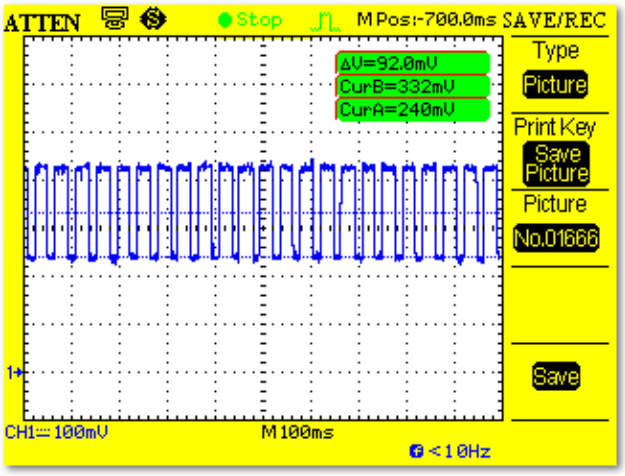


Fig. C.140.b. Proceso durante el escaneo

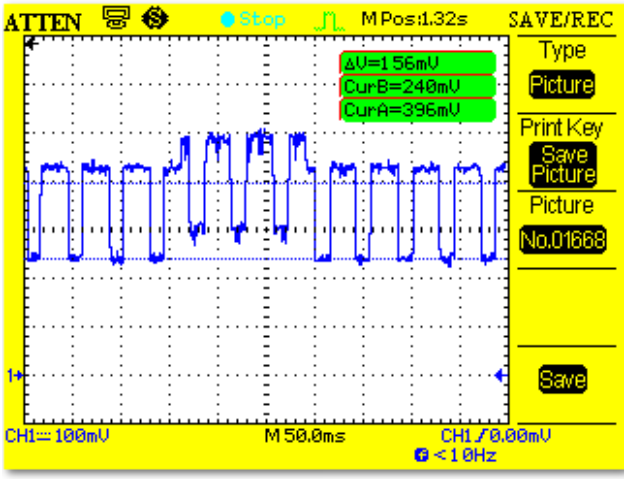


Fig. C.140.c. Detección de dispositivo BLE

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	5260	9.03	47520

Tabla C.146. Consumo del proceso total del escaneo de dispositivos BLE

C.3.3.3. Escaneo del nombre del dispositivo

Este ejemplo muestra cómo hacer un escaneo del nombre del dispositivo con el BLE a la máxima potencia, imprimiendo el número de dispositivos descubiertos y escanear los resultados guardados en la EEPROM.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

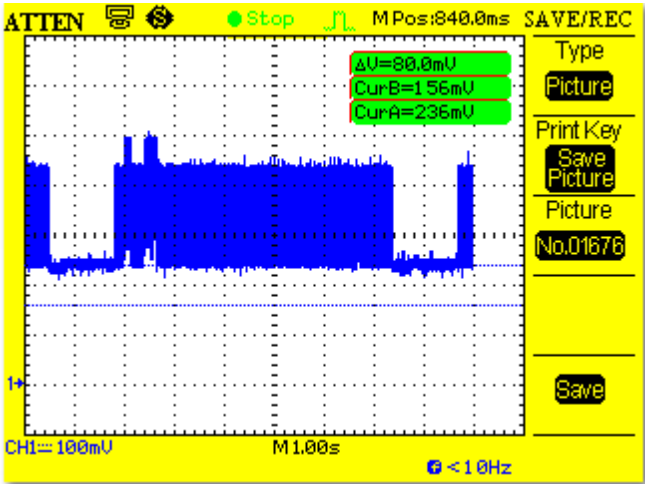


Fig. C.141.a. Medida general del proceso

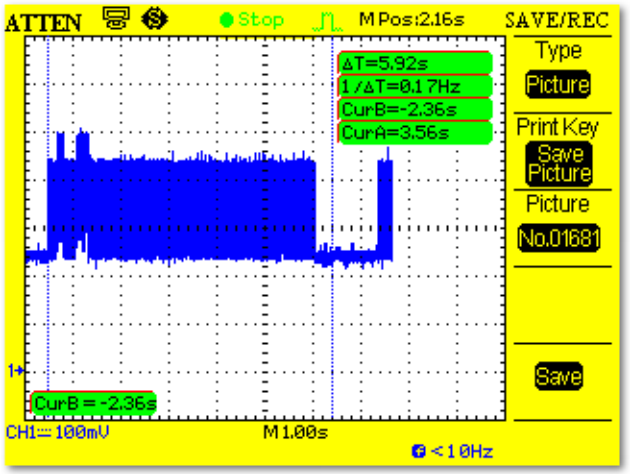


Fig. C.141.b. Tiempo total del proceso

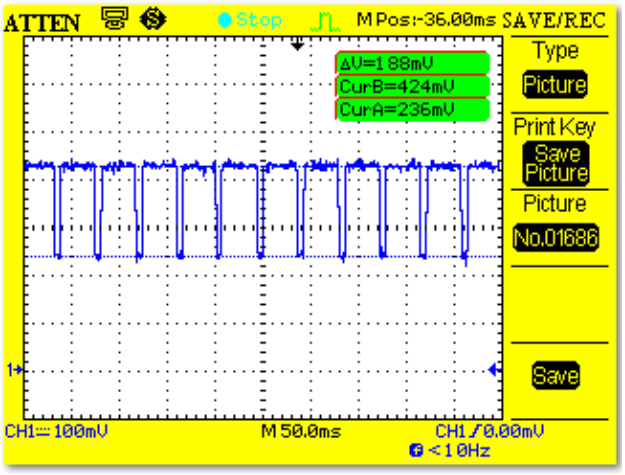


Fig. C.141.c. Proceso durante el escaneo BLE

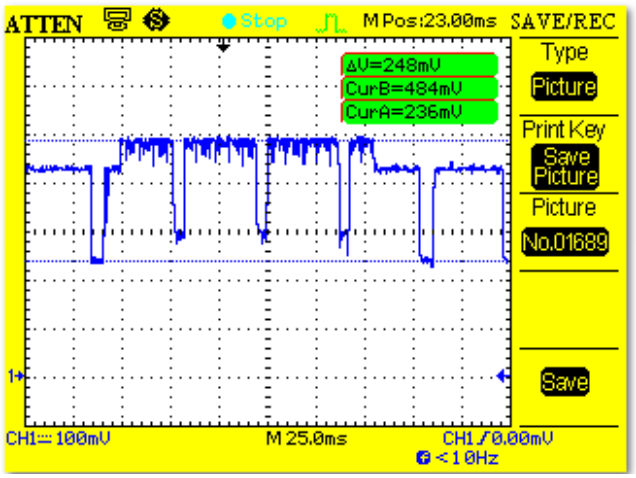


Fig. C.141.d. Detección de dispositivo

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	5920	12.32	72986.2

Tabla C.147. Consumo del proceso total del escaneo de dispositivos BLE

C.3.3.4. Escaneo limitado

Este ejemplo muestra cómo hacer un escaneo limitado con el módulo BLE a máxima potencia, imprimiendo el número de dispositivos descubiertos con y guardándolos en la EEPROM.

Se puso como límite para que se descubrieran dos dispositivos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

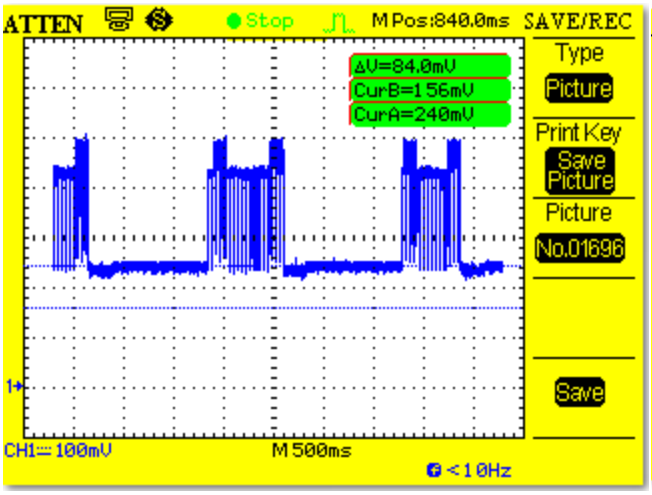


Fig. C.142.a. Medida general del proceso

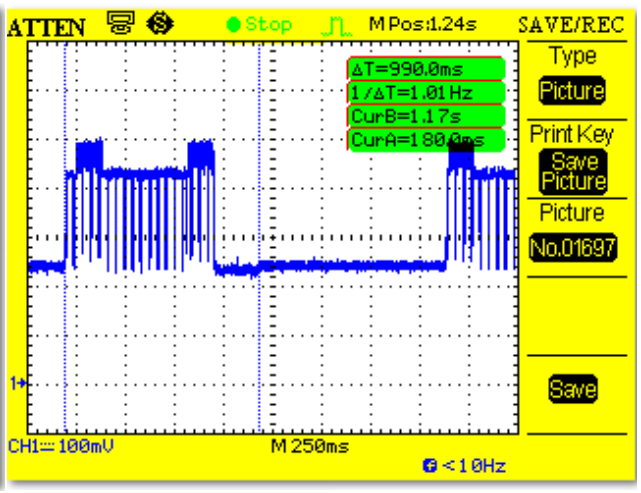


Fig. C.142.b. Tiempo total del proceso

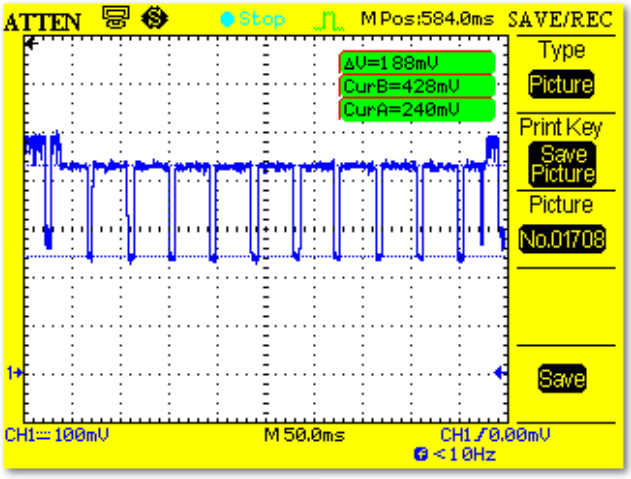


Fig. C.142.c. Proceso durante el escaneo

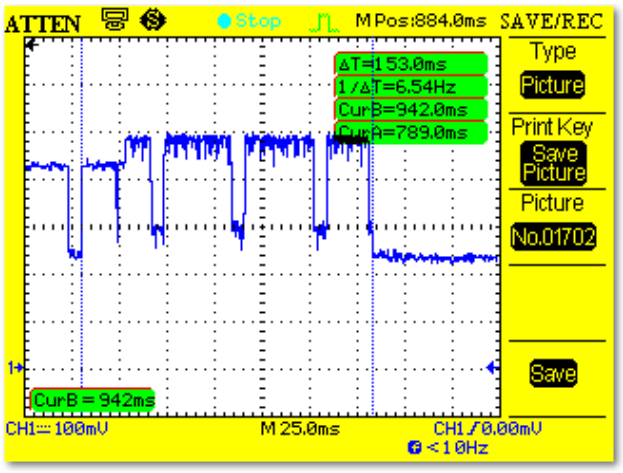


Fig. C.142.d. Detección de dispositivo BLE

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	990	11.21	11102.84

Tabla C.148. Consumo del proceso total del escaneo de dispositivos BLE

C.3.3.5. – Escaneo del dispositivo

Este ejemplo muestra cómo hacer un escaneo con el BLE a máxima potencia y buscar un modulo BLE específico. Se pone la dirección MAC del dispositivo a encontrar.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

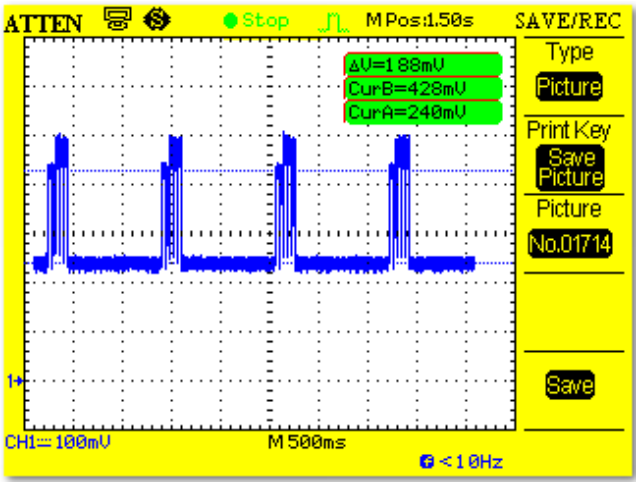


Fig. C.143.a. Medida general del proceso

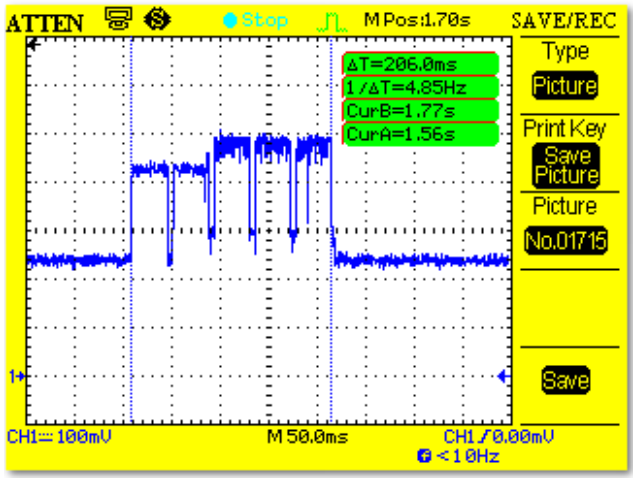


Fig. C.143.b. Tiempo total del proceso de escaneo

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	206	20.33	4189.6

Tabla C.149. Consumo del proceso total del escaneo de dispositivos BLE

C.3.3.6. – Configurar una conexión

Este ejemplo configure los parámetros involucrados cuando se conecta a un esclavo. Representa un escaneo del dispositivo y si lo detecta, se conecta a un módulo BLE con los parámetros seleccionados. Para ello, pondremos la dirección MAC del dispositivo a detectar.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

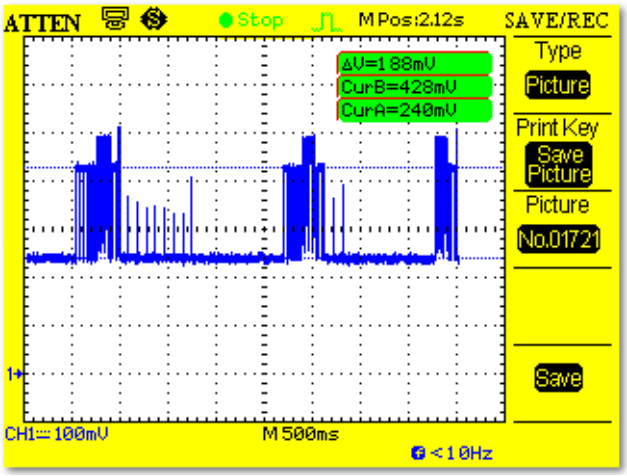


Fig. C.144.a. Medida general del proceso

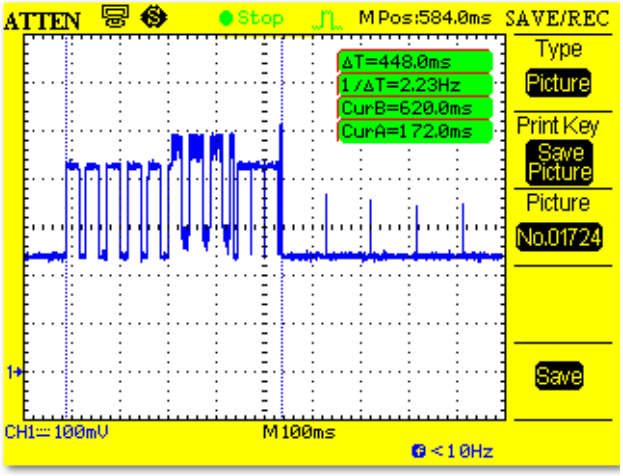


Fig. C.144.b. Tiempo total del proceso de escaneo

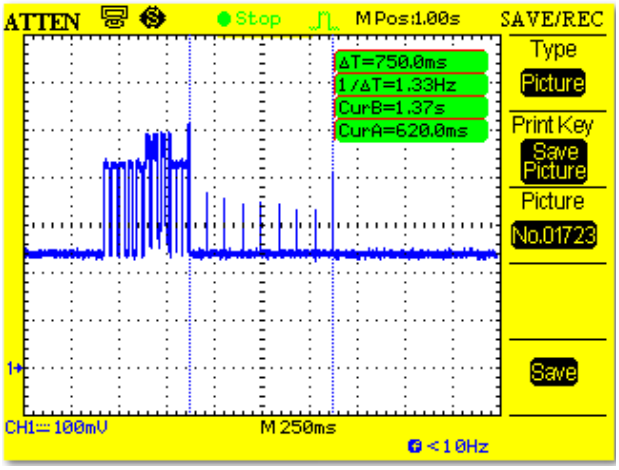


Fig. C.144c. Tiempo de conexión con el dispositivo

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	1198	5.5	6598.8

Tabla C.150. Consumo del proceso total para establecer conexión con el dispositivo

C.3.3.7. Configurar un escaneo

Este ejemplo muestra como configurar los parámetros de escaneo y hacer un escaneo con ellos, imprimiendo el número de dispositivos descubiertos y escanear los resultados guardados en la EEPROM.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

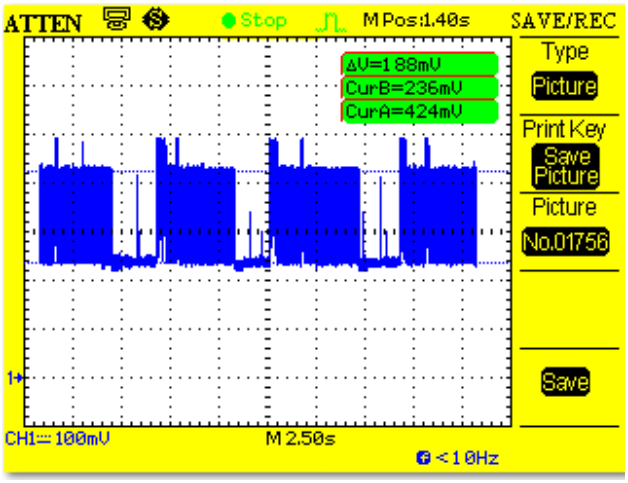


Fig. C.145.a Medida general del proceso

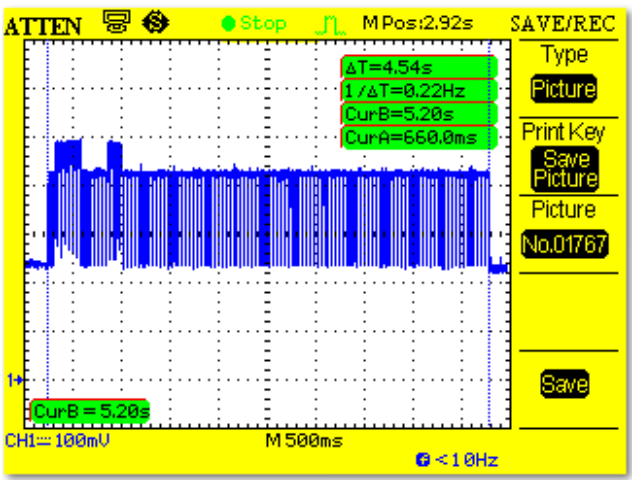


Fig. C.145.b. Tiempo total del proceso de escaneo

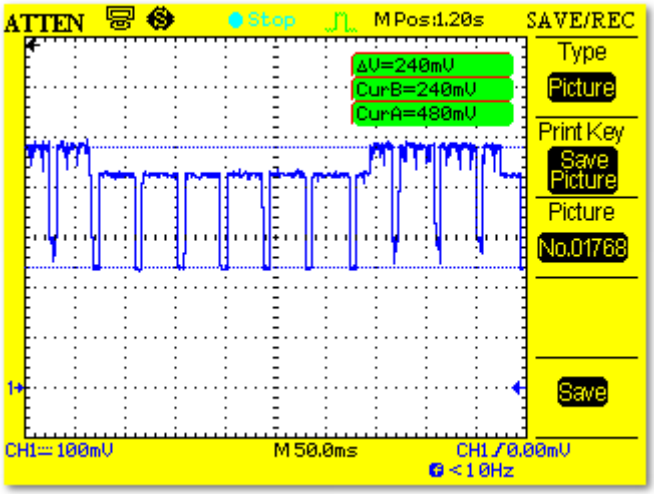


Fig. C.145.c. Proceso durante el escaneo con detección de dispositivos BLE

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	5020	12.19	61202.4

Tabla C.151. Consumo del proceso total de configuración del escaneo

C.3.3.8. Configurar avisos

Este ejemplo cambia los parámetros de avisos para mostrar la capacidad de envío de datos en la carga útil del aviso. Hay que recordar que la longitud máxima de datos de aviso es de 31 bytes.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

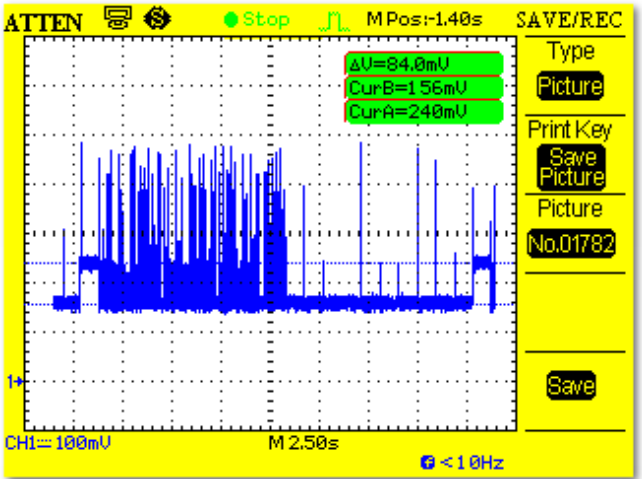


Fig. C.146.a. Medida general del proceso

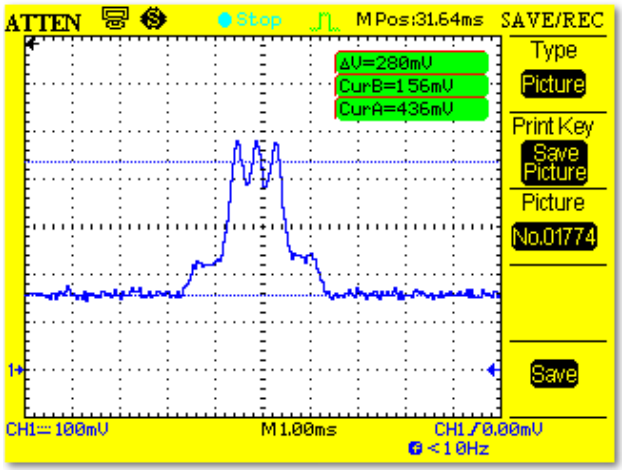


Fig. C.146.b. Avisos en 3 canales

Como se observa en las gráficas, se produce una serie de avisos en los 3 canales cada 100 ms durante 10 segundos y cada segundos durante 10 segundos.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	21000	0.62	1302

Tabla C.152. Consumo del proceso total de configuración del escaneo

C.3.3.9. – Conexión a un dispositivo BLE como maestro.

Este ejemplo muestra como conectarse a otro dispositivo BLE usando como estándar los parámetros de conexión. Este dispositivo será el maestro y el dispositivo remoto será el esclavo. Se lee o se escribe un atributo remoto en el esclavo. Para ello, pondremos la dirección MAC del dispositivo a detectar.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

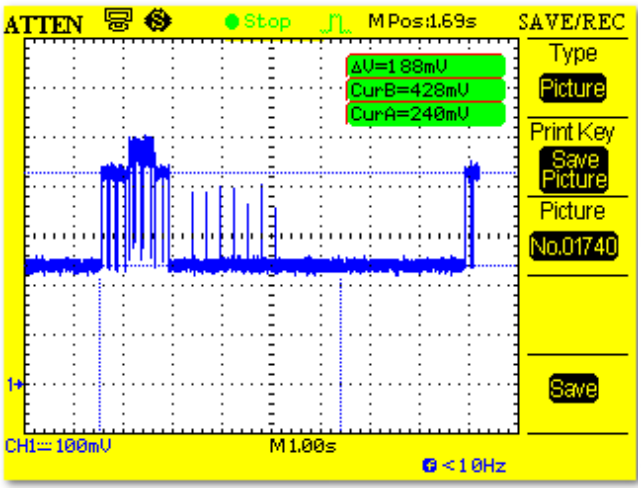


Fig. C.147.a. Medida general del proceso

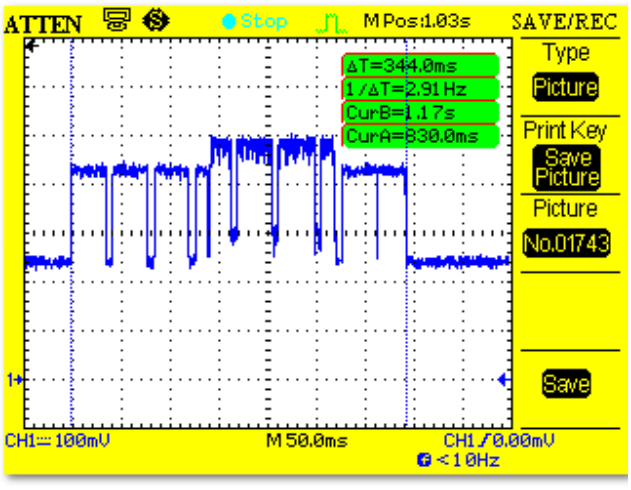


Fig. C.147.b. Tiempo del proceso de escaneo

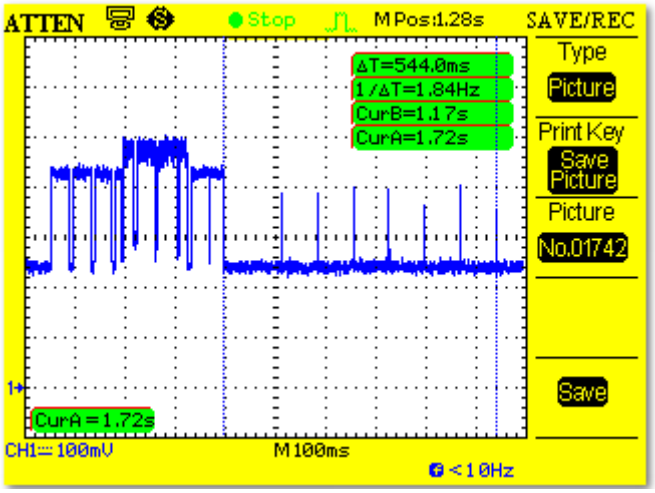


Fig. C.147.c. Tiempo de conexión con el dispositivo

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	888	7.34	6522.88

Tabla C.153. Consumo del proceso total de conexión a un dispositivo BLE como maestro

C.3.4. Módulo GPS

C.3.4.1. – Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo GPS cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

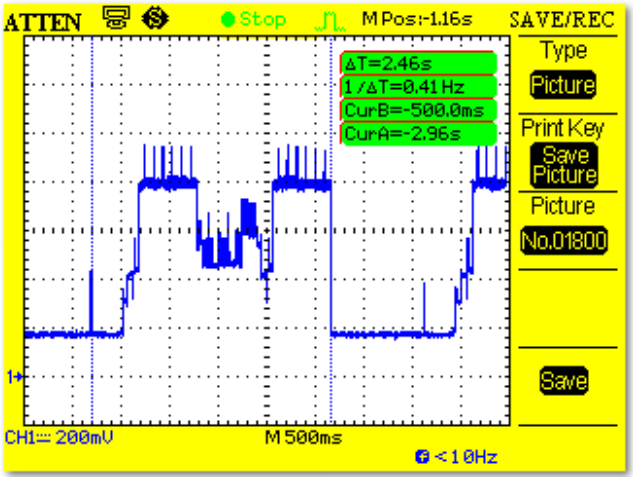


Fig. C.148.a. Medida general del proceso

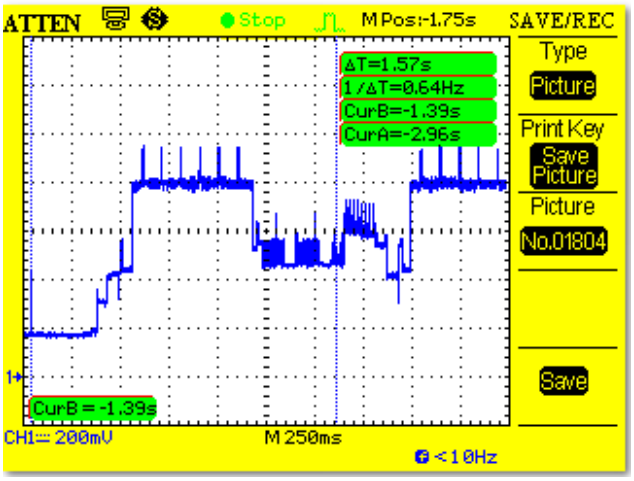


Fig. C.148.b. Proceso de encendido del módulo

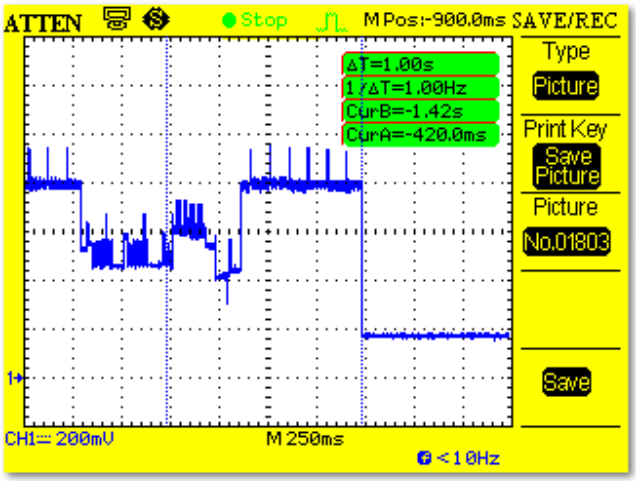


Fig. C.148.c. Estado on del módulo durante un segundo

Como se ve en la gráfica (b), el tiempo de ejecución del proceso de encendido del módulo es de 1570 ms, mientras que el proceso de apagado se realiza instantáneamente.

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso On	1570	36.48	57282.4
Estado On	1000	54.925	54925.4
Proceso Off	0	54.925	0

Tabla C.154. Consumo del proceso y estado on y del proceso off del módulo GPS

C.3.4.2. – Leer datos básicos desde el módulo GPS

Este ejemplo muestra como leer los datos básicos desde el módulo GPS. Para ello, hace falta esperar hasta que se conecte a los satélites, que normalmente, con buena cobertura, se conecta el módulo en menos de un minuto. A continuación, se solicita la información. Todos los datos se guardan en los atributos del GPS.

Esperar hasta que se conecta a los satélites. A continuación, se solicita información. Todos los datos se almacenan en los atributos de GPS. Finalmente, se hace una conversión a grados para facilitar la búsqueda de las posiciones del GPS.

Como las pruebas se realizaron al aire libre, en el código, se añadieron una serie de funciones para saber con seguridad si el módulo se conecta o no. Si el led 0 se enciende y se apaga es porque se ha llegado a conectar el módulo y si el Led 1 permanece todo el rato encendiéndose y apagándose es porque no hay conexión. Normalmente el módulo se conecta en menos de un minuto. Si tarda más de 240 segundos, que es el rango límite de conexión, es que no se llega a conectar. En algunas ocasiones hay que darle tiempo a que se conecte. No siempre pasa en menos de un minuto.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

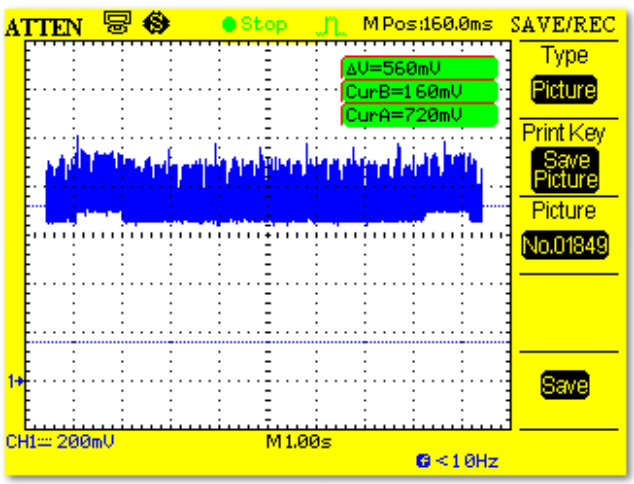


Fig. C.149.a. Proceso de la lectura de la posición del GPS

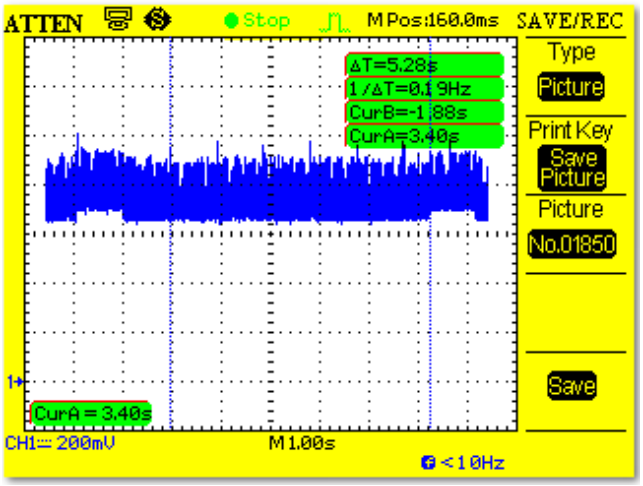


Fig. C.149.b. Tiempo de ejecución de la lectura de la posición

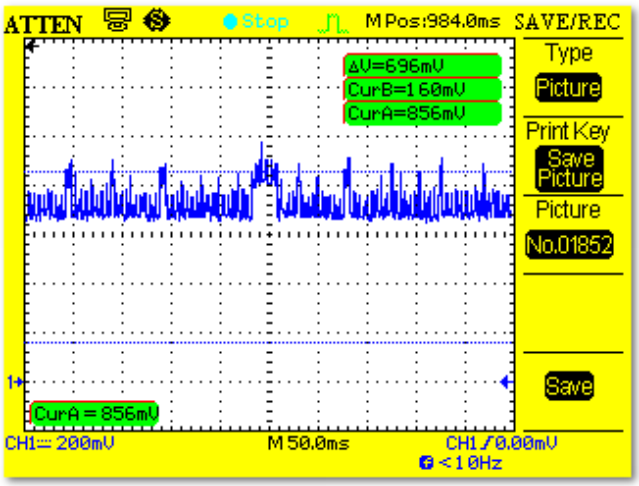


Fig. C.149.c. Lectura de cada dato

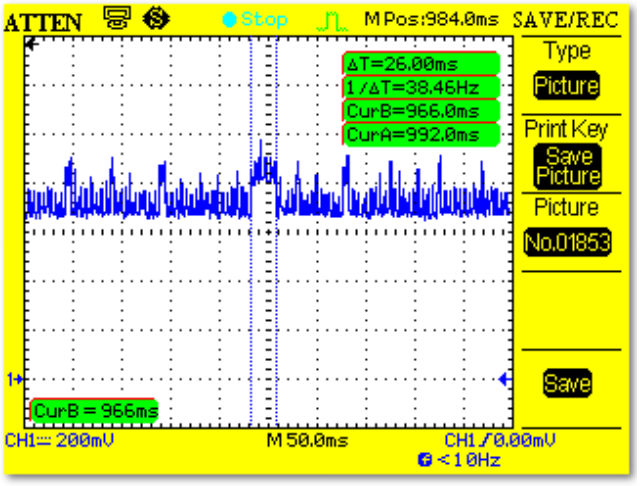


Fig. C.149.d. Tiempo de ejecución de la lectura de cada dato

Cabe destacar que, para el cálculo del consumo, se ha tenido en cuenta el tiempo de espera de conexión, que ha sido en torno a 60 segundos (aunque no se refleje en las gráficas) y la lectura de la posición del GPS, que como se ve en la gráfica (b), son 5280 ms.

También se ve que no se produce un consumo extra durante el proceso conexión respecto del estado on del GPS, por lo tanto el consumo de la lectura de la posición del módulo será mínimo.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	65280	0.034	2277.6

Tabla C.155. Consumo del proceso total de lectura de la posición del GPS

C.3.5. Módulo Wi-Fi

C.3.5.1. – Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo Wi-Fi en el socket 0 cada segundo. Antes, se llama a la función resetValues para resetear la configuración del módulo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

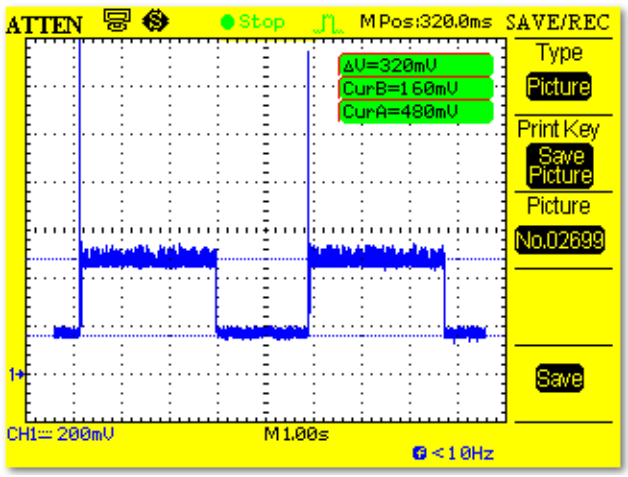


Fig. C.150.a. Medida general del proceso

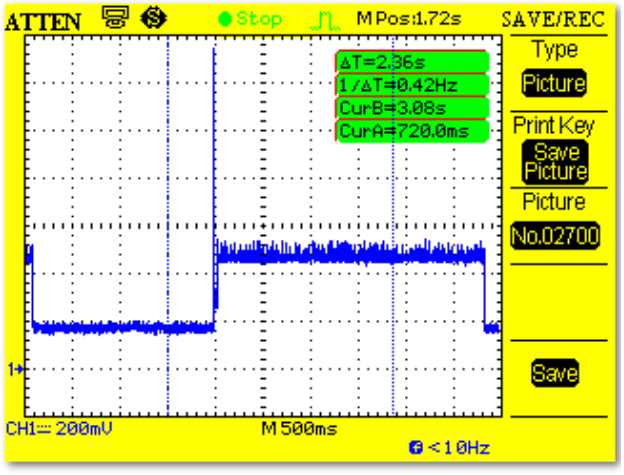


Fig. C.150.b. Proceso de encendido del módulo

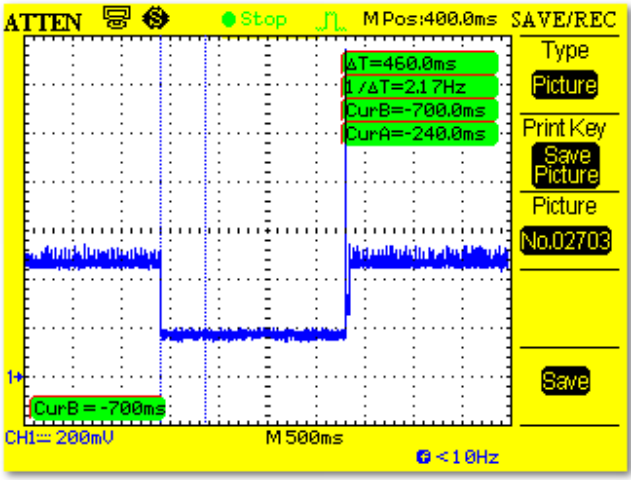


Fig. C.150.c. Proceso de apagado del módulo

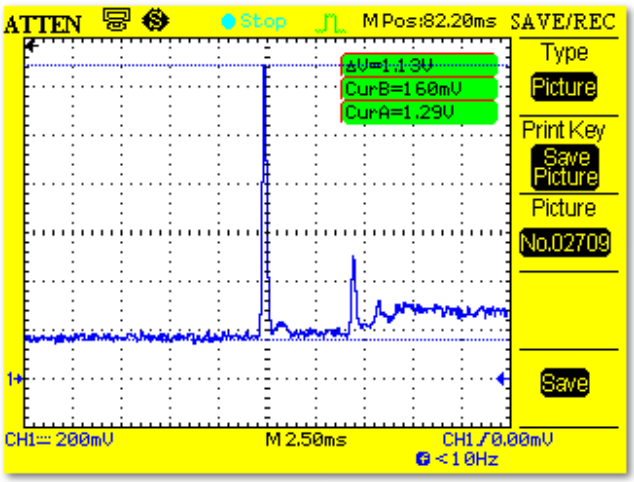


Fig. C.150.d. Pico de encendido

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso On	2360	25.91	61147.64
Estado On	1000	32.4	32400
Proceso Off	460	2	920

Tabla C.156. Consumo del proceso y estado on y del proceso off del módulo Wi-Fi

C.3.5.2. – Unión a un punto de acceso

Este ejemplo muestra cómo unirse a un punto de acceso cifrado WPA.

Para que funcione la conexión, se debe configurar, previamente, los parámetros AP. Hay que definir una cuenta Wifi y su contraseña correspondiente.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

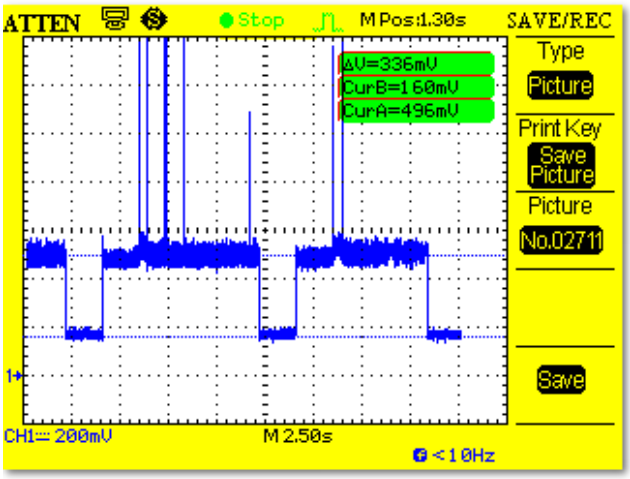


Fig. C.151.a. Medida general del proceso

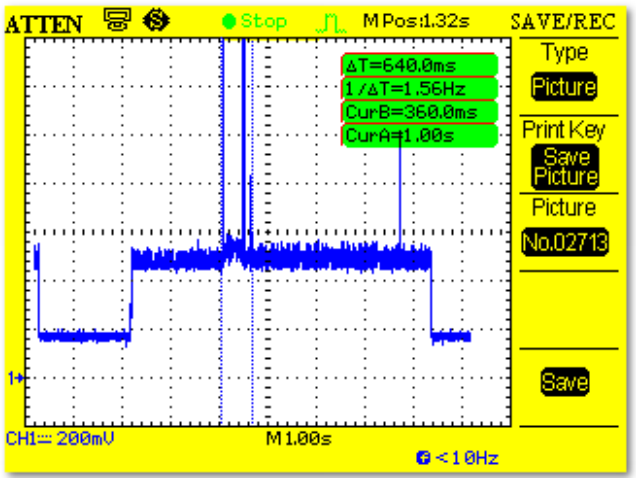


Fig. C.151.b. Proceso de unión

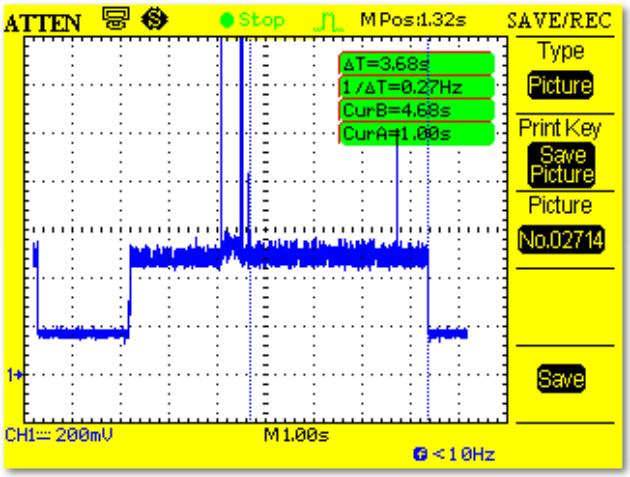


Fig. C.151.c. Procesos del estado del AP

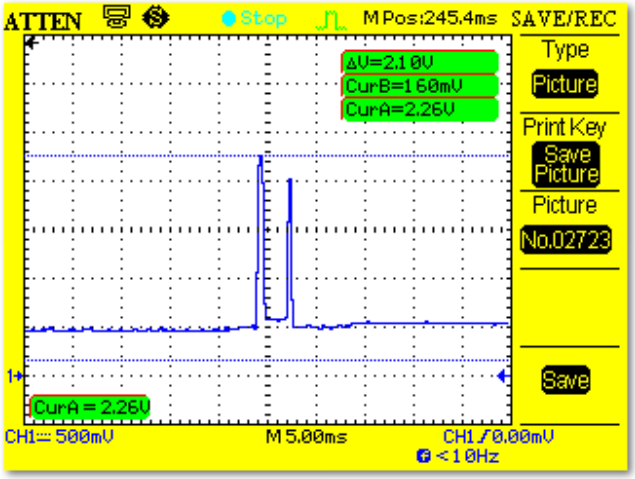


Fig. C.151.d. Picos producidos al unir

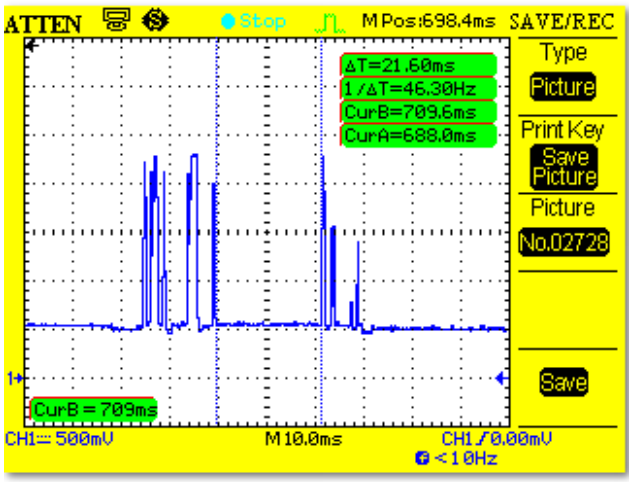


Fig. C.151.e. Picos al final del proceso de unión

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	7140	29.67	211833.16

Tabla C.157. Consumo del proceso total para la unión a un punto de acceso

C.3.5.3. – Envío de mensajes de solicitud HTTP

Este ejemplo muestra como enviar mensajes de solicitud HTTP. Libelium creó un servidor web para ello.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

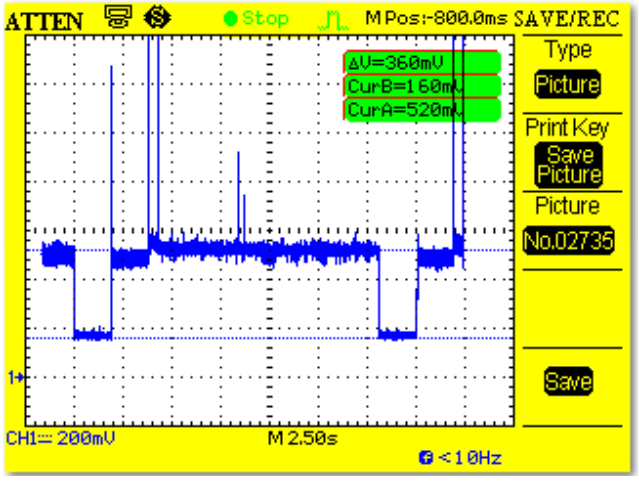


Fig. C.152.a. Medida general del proceso

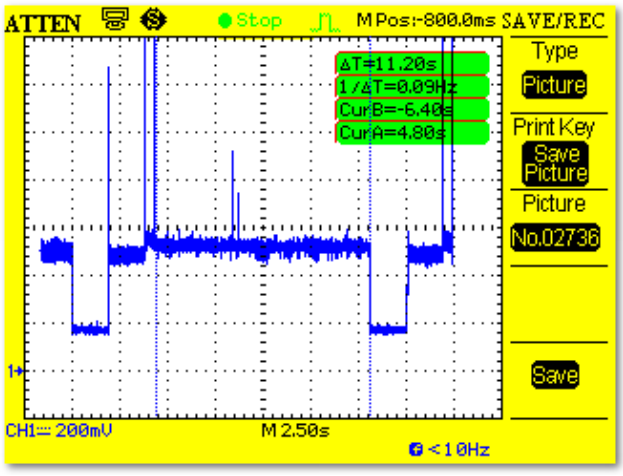


Fig. C.152.b. Proceso de envío de solicitud HTTP

Como se ve en la gráfica (b), el tiempo de ejecución del proceso de envío de mensajes de solicitud HTTP es de 11200 ms.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	14660	33.72	494443.16

Tabla C.158. Consumo del proceso total para el envío de mensajes de solicitud HTTP

C.3.5.4. – Envío de mensajes de solicitud HTTP usando tramas

Este ejemplo muestra como enviar mensajes de solicitud HTTP usando tramas. Libelium creó un servidor web para ello.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

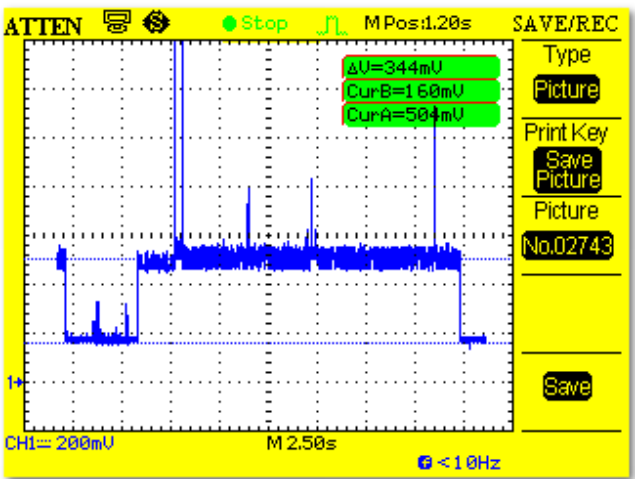


Fig. C.153.a. Medida general del proceso

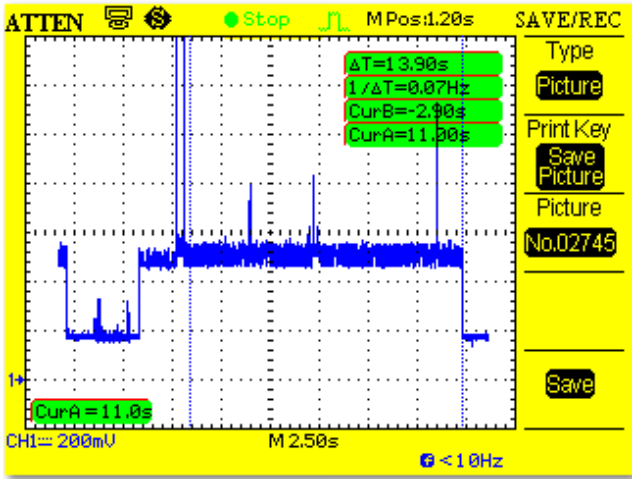


Fig. C.153.b. Proceso de envío de solicitud HTTP

Como se ve en la gráfica (b), el tiempo de ejecución del proceso de envío de mensajes de solicitud HTTP es de 13900 ms.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	17360	32.86	570483.16

Tabla C.159. Consumo del proceso total para el envío de mensajes de solicitud HTTP

C.3.5.5. – Conexión de cliente TCP

Este ejemplo muestra cómo crear una conexión de cliente TCP usando tramas. Para ello habrá que configurar el servidor TCP, poniendo la correspondiente dirección IP, puerto remoto y puerto local. Con respecto al tamaño de las tramas, se pondrá un valor cercano al máximo, es decir, 130 bytes, añadiendo los bytes necesarios para tal fin.

Para este ejemplo, las funciones de encendido del módulo y de unión a un punto de acceso se ejecutarán en el setup.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

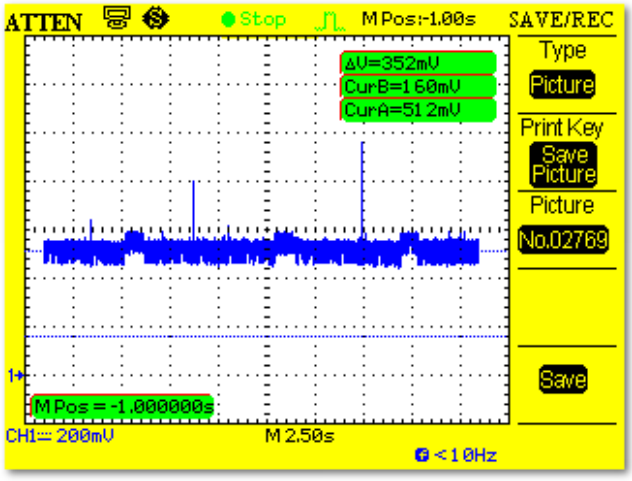


Fig. C.154.a. Medida general del proceso

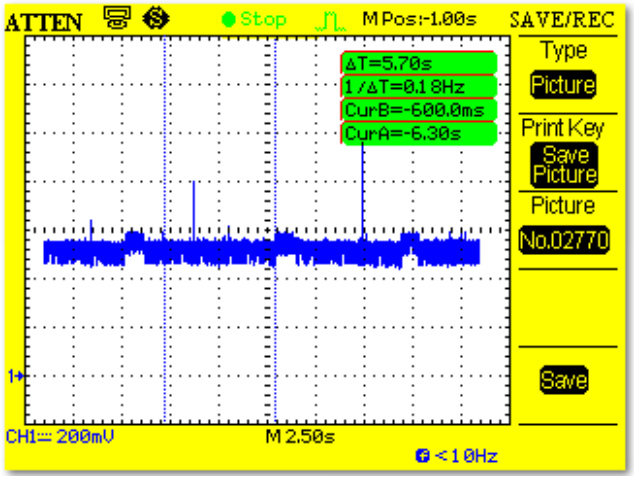


Fig. C.154.b. Tiempo de ejecución de la conexión TCP

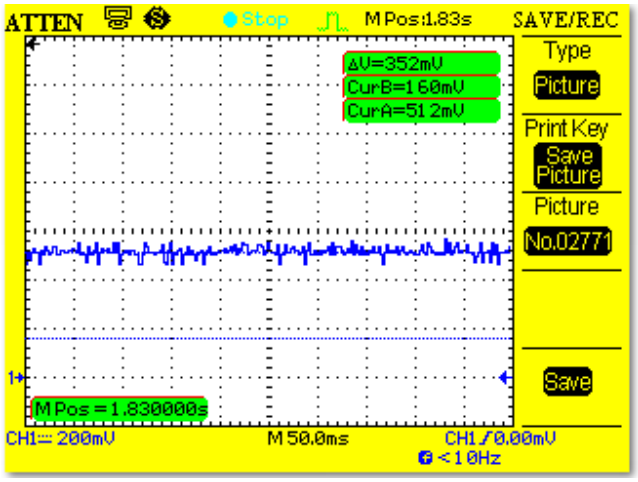


Fig. C.154.c. Proceso de conexión TCP

Como se ve en las gráficas, este tipo de acción no produce un consumo extra después de

realizar la conexión a un punto de acceso, por eso sale la señal continua. Con lo cual, para saber cuánto dura el tiempo de ejecución de dicha acción, se añaden una serie de funciones que fuerza un consumo extra en el proceso y así se puede distinguir mejor cuando dura ejecutarlo, como se ve en la gráfica (b).

El consumo del proceso total es el siguiente:

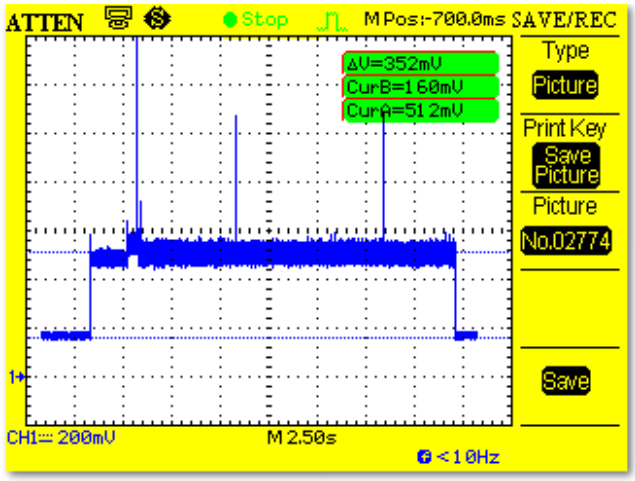
	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	5700	0	0

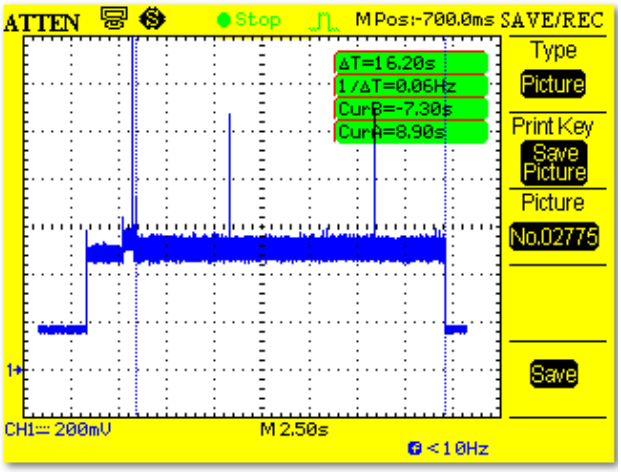
Tabla C.160. Consumo del proceso total de conexión TCP

C.3.5.6. – Conexión de cliente UDP

Este ejemplo muestra cómo crear una conexión de cliente UDP. Para ello habrá que configurar el servidor UDP, poniendo la correspondiente dirección IP, puerto remoto y puerto local.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:





El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	19660	33.74	663483.17

Tabla C.161. Consumo del proceso total de conexión UDP

C.3.6. Módulo GPRS

C.3.6.1. – Proceso de encendido y apagado del módulo

Este ejemplo muestra como encender y apagar el módulo GPRS cada segundo.
A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

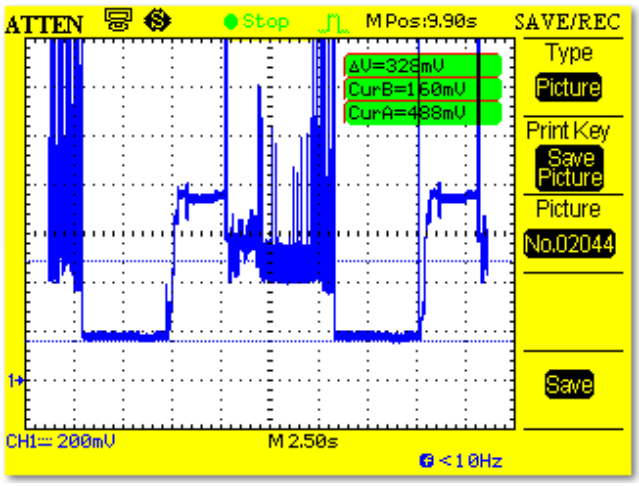


Fig. C.155.a. Medida general del proceso

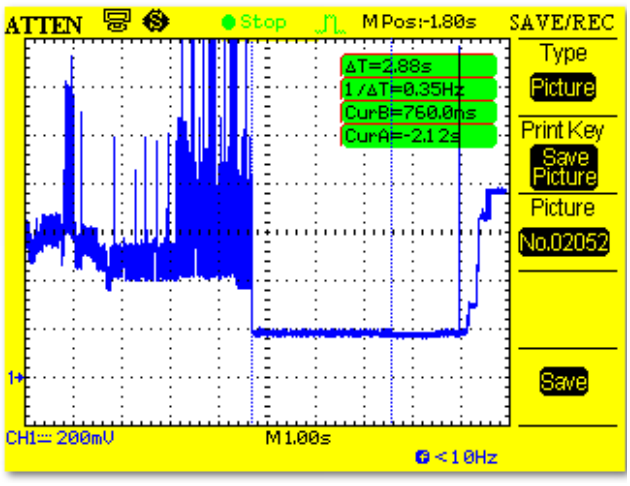


Fig. C.155.b. Proceso de apagado del módulo

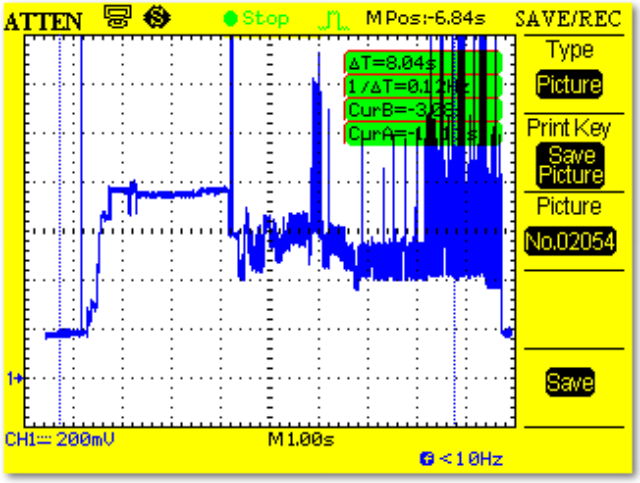


Fig. C.155.c. Proceso de encendido del módulo

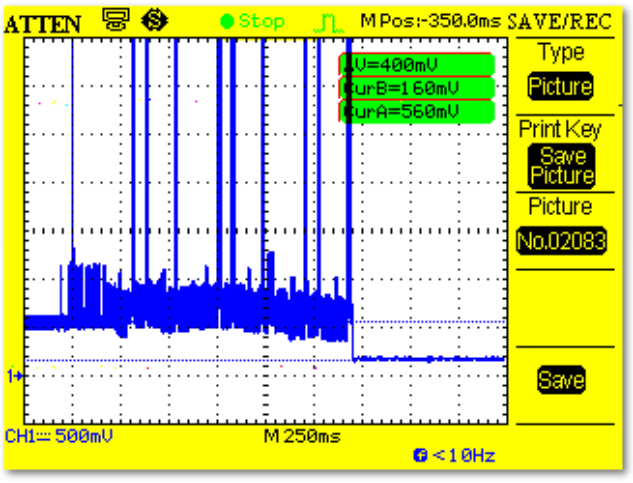


Fig. C.155.d. Proceso durante el estado on

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso On	8040	43.48	349589.88
Estado On	1000	24.4	24400
Proceso Off	2880	2.8	6064

Tabla C.162. Consumo del proceso y estado on y del proceso off del módulo GPRS

C.3.6.2. – Proceso de conexión a la red

Este ejemplo muestra como conectarse a la red GPRS. El tiempo de conexión puede variar, aunque, normalmente, con buena señal, se conecta en menos de 10 segundos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

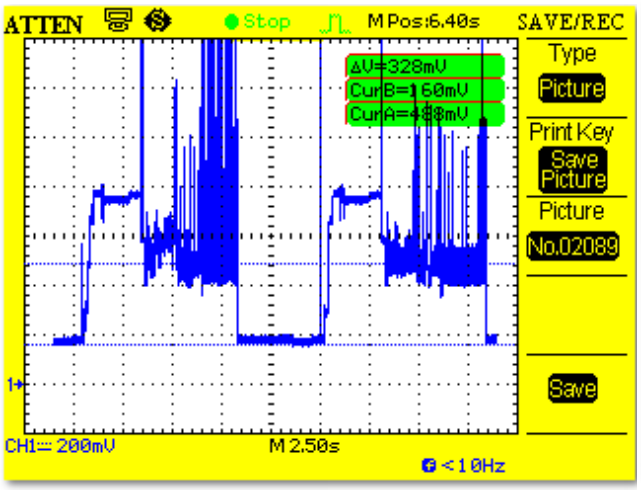


Fig. C.156.a. Medida general del proceso

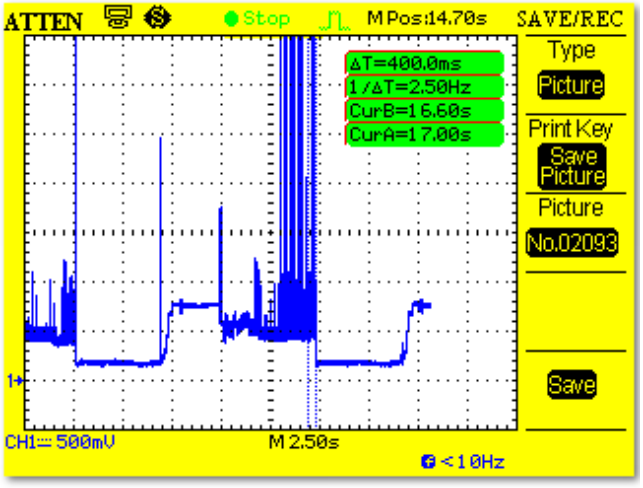


Fig. C.156.b. Tiempo de conexión

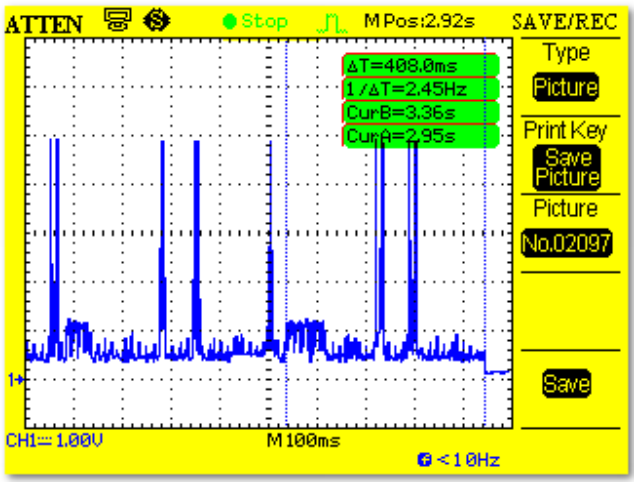


Fig. C.156.c. Proceso de conexión a la red

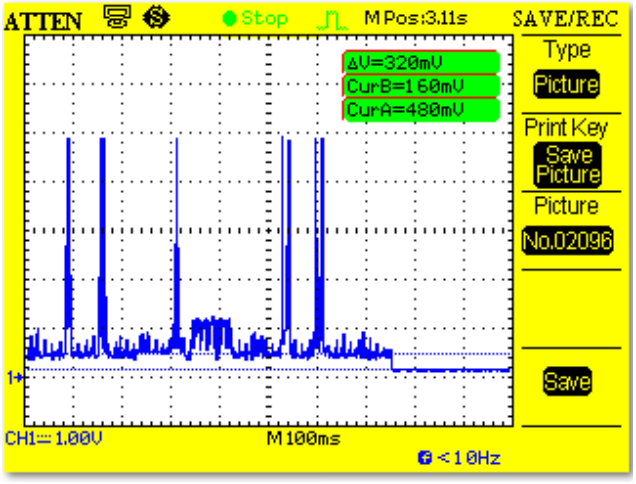


Fig. C.156.d. Consumo proceso de conexión

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	11320	32.28	365413.88

Tabla C.163. Consumo del proceso total para conectar el módulo a la red

C.3.6.3. – Envío de mensajes (SMS)

Este ejemplo muestra como enviar un mensaje. Hay que poner el número de teléfono de destino. Para realizar mejor las medidas, se ha dejado las funciones de encendido y de conexión en el setup.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

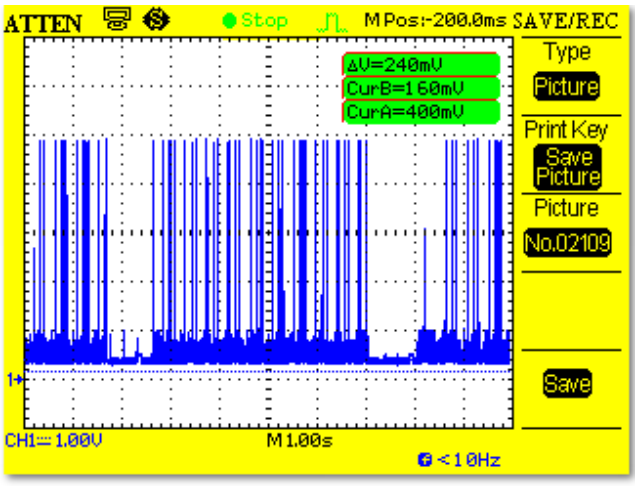


Fig. C.157.a. Medida general del proceso

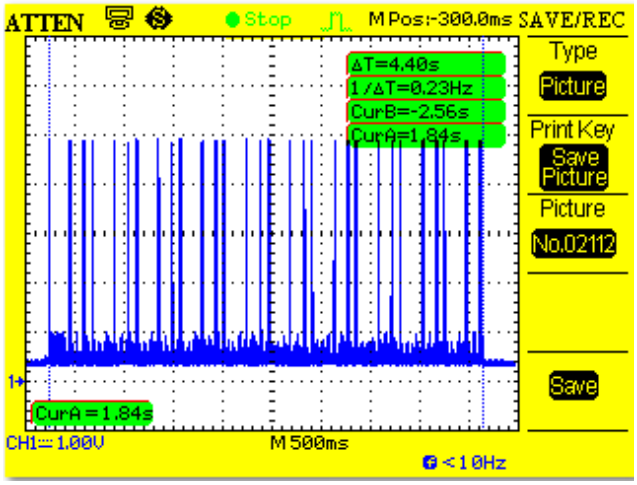


Fig. C.157.b. Tiempo de envio del sms

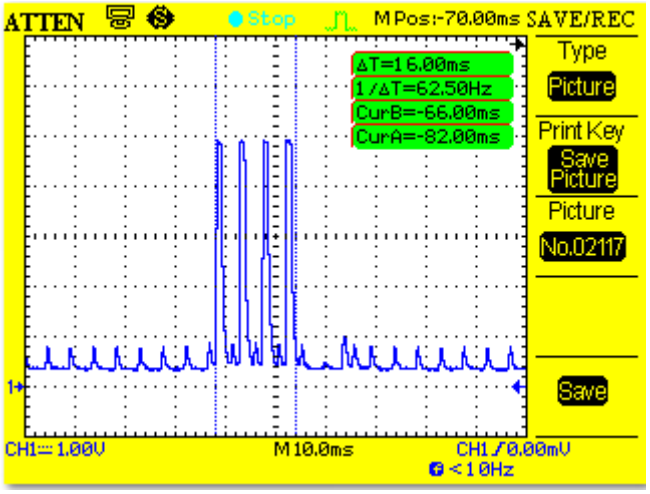


Fig. C.157.c. Anchura pulsos

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	4400	24.43	107520

Tabla C.164. Consumo del proceso del envío de SMS

C.3.6.4. – Lectura de una URL con petición GET

Este ejemplo muestra cómo leer una URL con una petición GET incluyendo una trama en la petición GET. Libelium creó un servidor web para ello. Con respecto al tamaño de las tramas, se pondrá un valor cercano al máximo, es decir, 130 bytes, añadiendo los bytes necesarios para tal fin.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

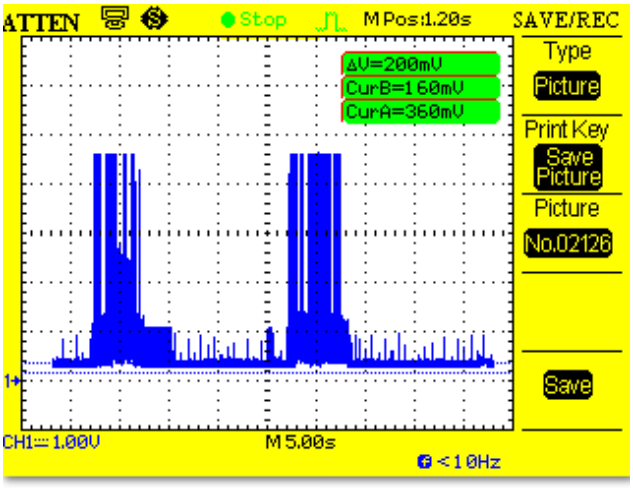


Fig. C.158.a. Medida general del proceso

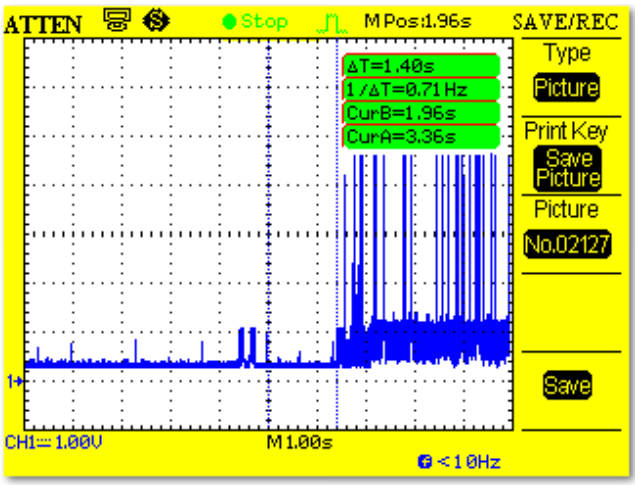


Fig. C.158.b. Proceso de configuración HTTP

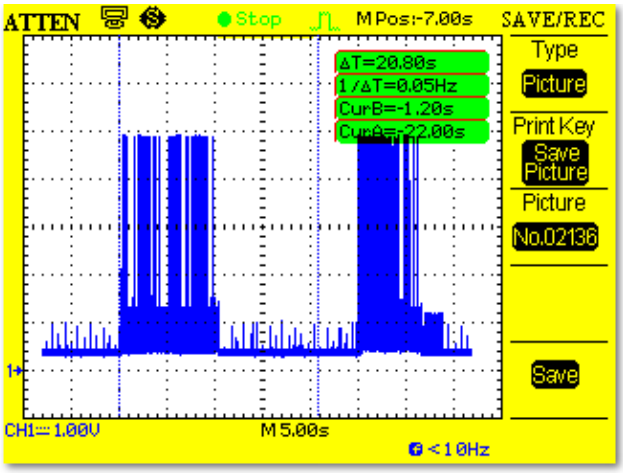


Fig. C.158.c. Tiempo de ejecución del proceso de lectura de la URL

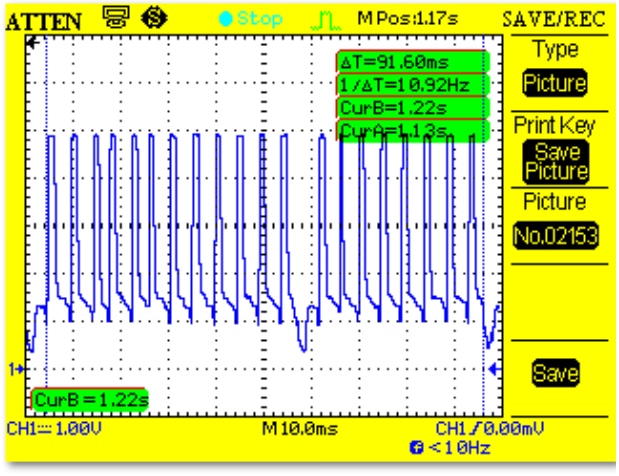


Fig. C.158.d. Anchura de los pulsos

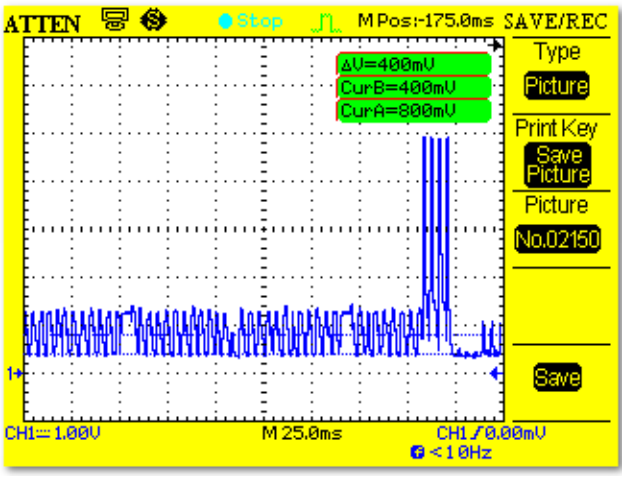


Fig. C.158.e. Proceso de lectura de la URL

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	24040	23.55	566144

Tabla C.165. Consumo del proceso de la lectura de una URL enviándose tramas

C.3.6.5. – Subir archivos al servidor FTP

Este ejemplo muestra como subir un archivo al servidor FTP creado por Libelium.

Para ello, es necesario la utilización de la tarjeta de memoria SD. Se crea un archivo del tamaño que se quiera. En este caso, se puso 300 bytes y se sube al servidor.

Es necesario configurar una serie de parámetros para tal fin. Se deberá de poner el correspondiente nombre de usuario, contraseña, servidor IP y el puerto del servidor FTP.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

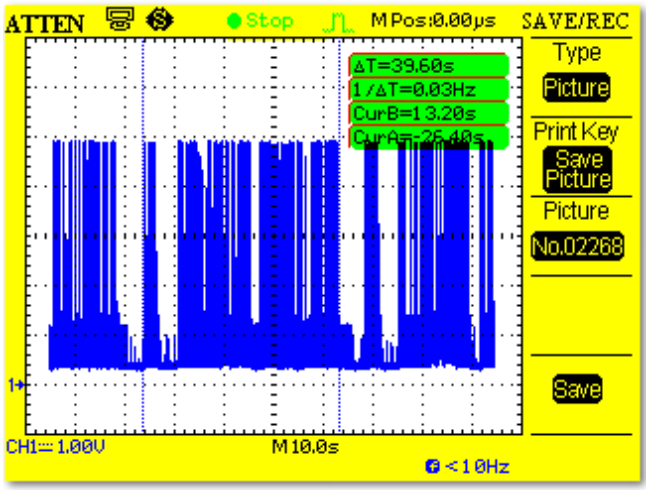


Fig. C.159.a. Tiempo de ejecución del proceso de la subida de archivo al servidor

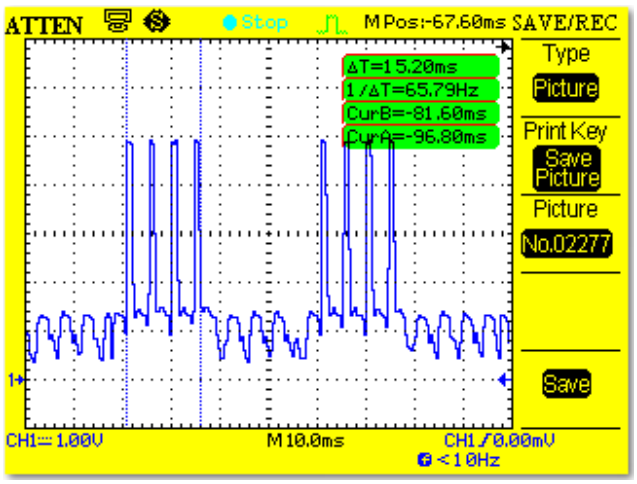


Fig. C.159.b. Anchura de los pulsos

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	41000	91.17	3738240

Tabla C.166. Consumo del proceso de subida de archivos al servidor FTP

C.3.6.6. – Conexión TCP

Este ejemplo muestra cómo crear un cliente TCP no transparente en una sola conexión. Se realizará el envío de datos usando tramas.

Para ello, habrá que configurar el servidor TCP, poniendo la correspondiente dirección IP y el puerto.

Con respecto al tamaño de las tramas, se pondrá un valor cercano al máximo, es decir, 130 bytes, añadiendo los bytes necesarios para tal fin.

Para este ejemplo, se midió el proceso completo: encendido del modulo, conexión a la red, configuración de la conexión TCP, conexión TCP, envío de datos y apagado del módulo por dar errores al realizar esta acción dejando el módulo encendido todo el rato.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

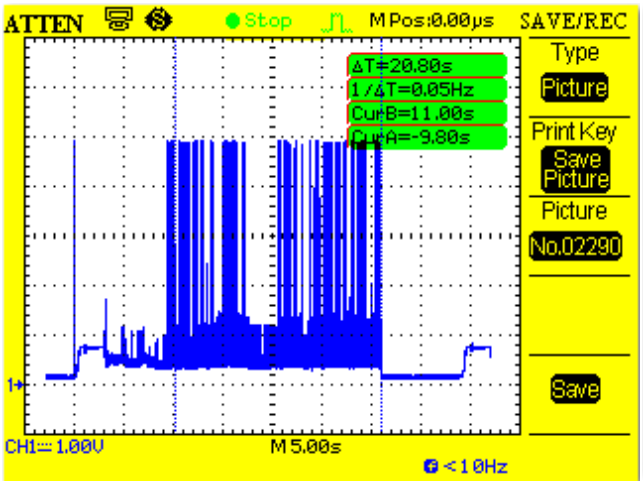


Fig. C.160.a. Tiempo de ejecución del proceso de conexión TCP

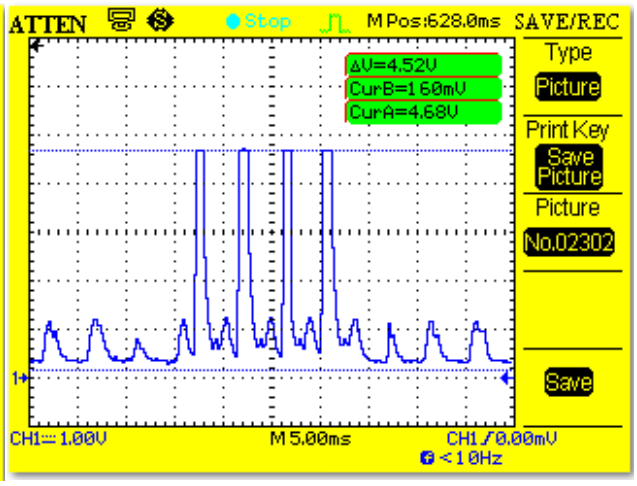


Fig. C.160.b. Pulsos durante el proceso de conexión TCP

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	32120	69.93	2246449.88

Tabla C.167. Consumo del proceso total para establecer conexiones TCP

C.3.6.7. – Conexión UDP

Este ejemplo muestra cómo crear un cliente UDP no transparente en una sola conexión. Se realizará el envío de datos usando tramas.

Para ello, habrá que configurar el servidor UDP, poniendo la correspondiente dirección IP y el puerto.

Con respecto al tamaño de las tramas, se pondrá un valor cercano al máximo, es decir, 130 bytes, añadiendo los bytes necesarios para tal fin.

Para este ejemplo, se midió el proceso completo: encendido del modulo, conexión a la red, configuración de la conexión UDP, conexión UDP, envío de datos y apagado del módulo por dar errores al realizar esta acción dejando el módulo encendido todo el rato.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

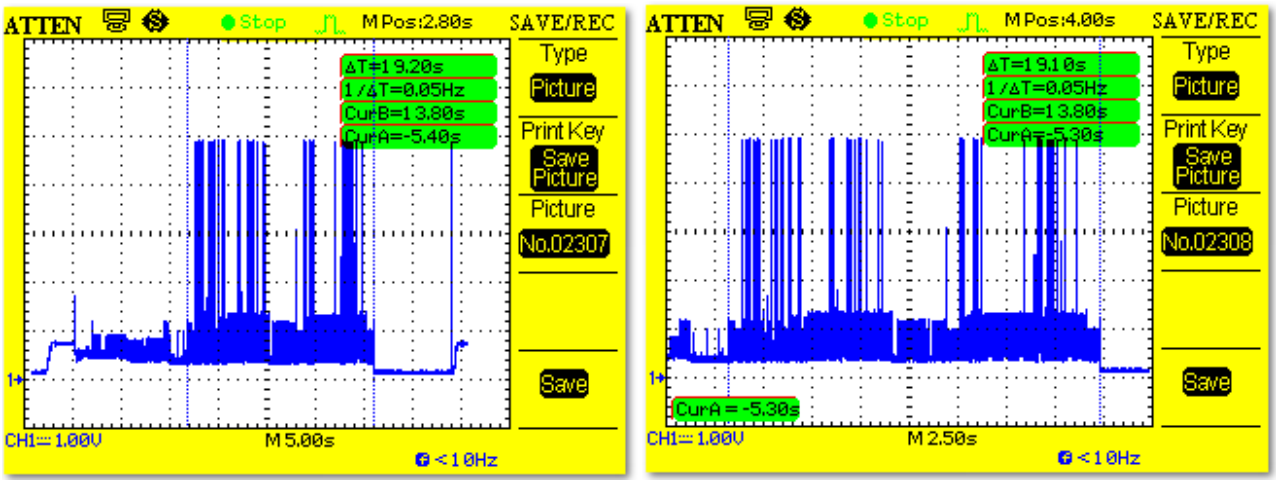


Fig. C.161.a. Tiempo de ejecución del proceso de conexión UDP

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	30520	57.47	1753989.88

Tabla C.168. Consumo del proceso total para establecer conexiones UDP

C.3.7. Módulo 3G

C.3.7.1. – Proceso de encendido y apagado del módulo 3G

Este ejemplo muestra como encender y apagar el módulo 3G cada segundo.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

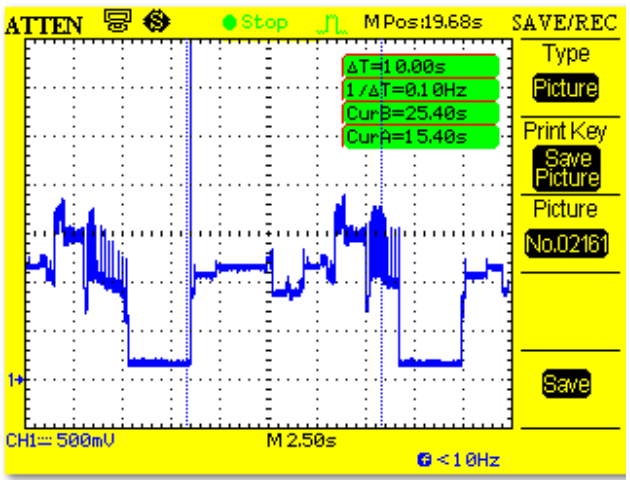


Fig. C.162.a. proceso on del módulo

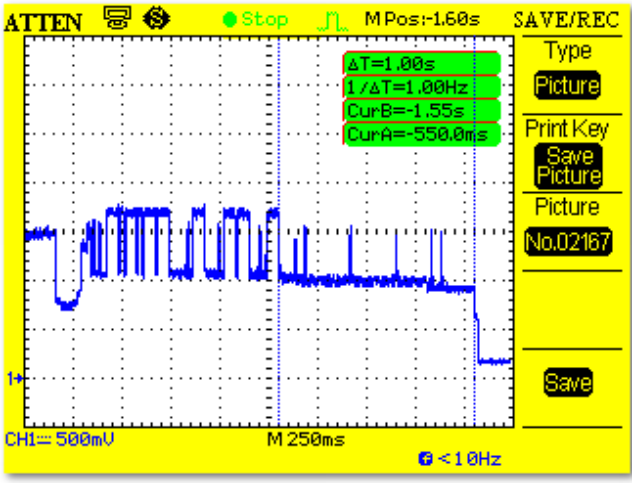


Fig. C.162.b. Estado on del módulo

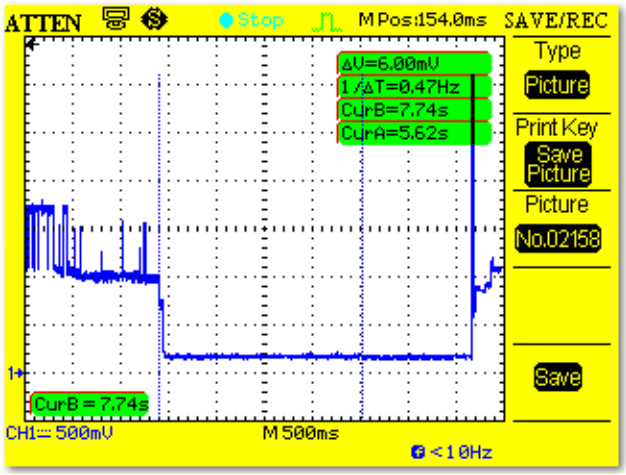


Fig. C.162.c. Proceso off del módulo

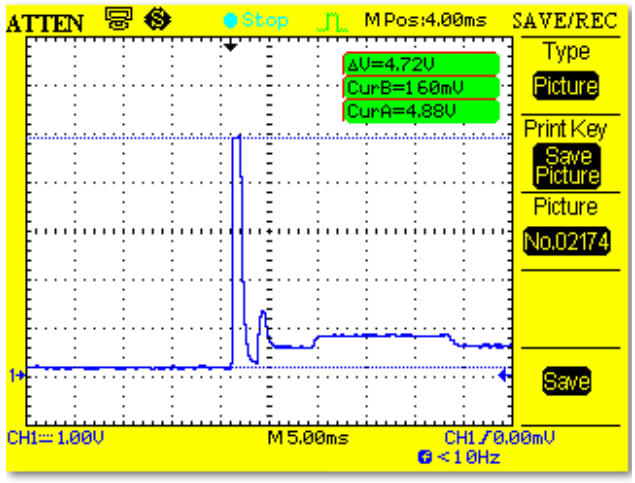


Fig. C.162.d. Pico de carga al encenderse el módulo

El consumo en el proceso y estado on y en el proceso off del módulo es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso On	10000	99.92	999237.56

Estado On	1000	82.4	82400
Proceso Off	2100	2.4	5040

Tabla C.169. Consumo del proceso y estado on y del proceso off del módulo 3G

C.3.7.2. – Proceso de conexión a la red

Este ejemplo muestra como conectarse a la red 3G. El tiempo de conexión puede variar, aunque, normalmente, con buena señal, se conecta en menos de 20 segundos.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

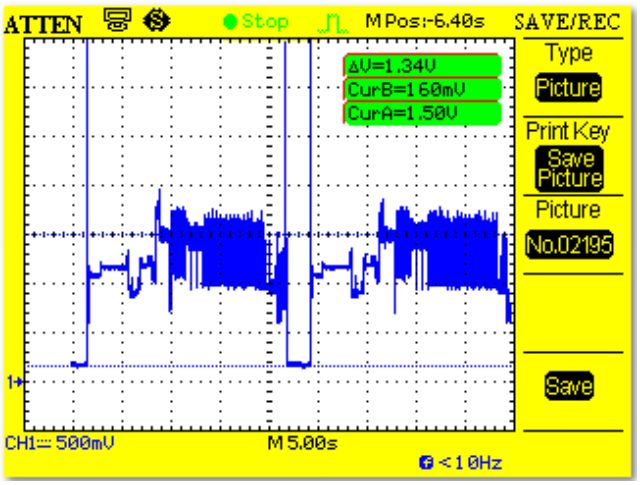


Fig. C.163.a. Medida general del proceso

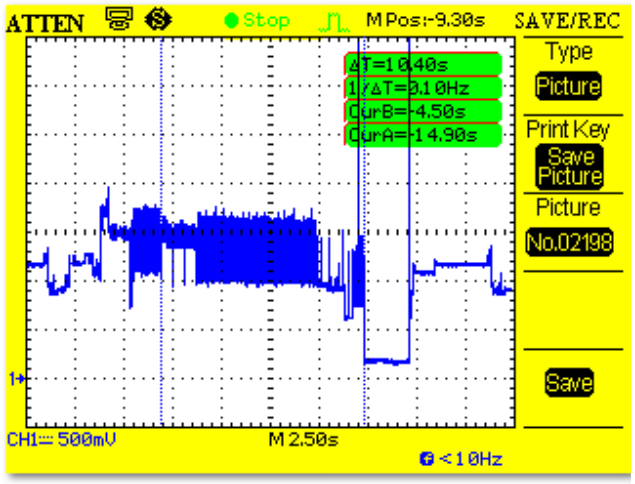


Fig. C.163.b. Tiempo de ejecución del proceso de conexión

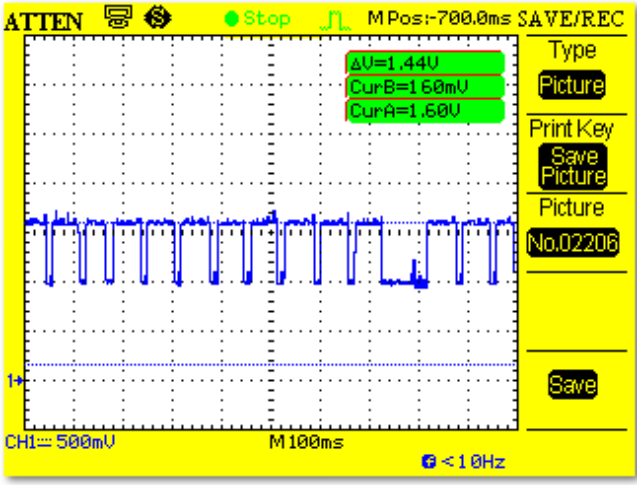


Fig. C.163.c. Consumo durante el proceso de conexión

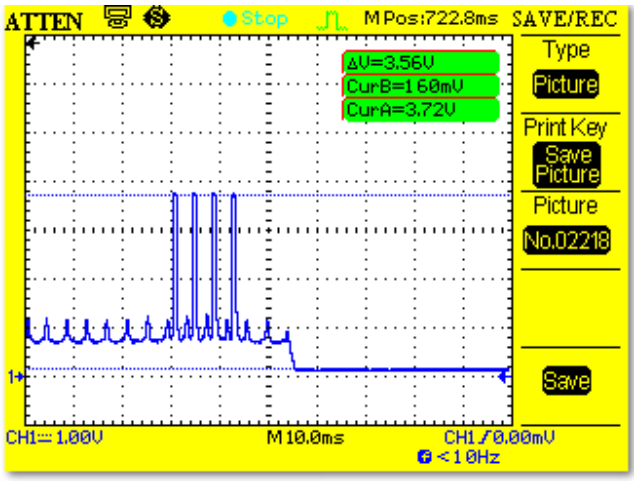


Fig. C.163.d. Proceso final del proceso de conexión

Se obtó, para la medición del proceso de conexión a la red del módulo, por realizar el proceso completo desde que se enciende el módulo hasta que se apaga ya que, si se deja, el On del módulo, en el setup(), le costaba menos conectarse de lo que normalmente se conecta, con lo cual, falsearía la medida.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	22500	101.8	2290516.69

Tabla C.170. Consumo del proceso total para establecer conexione a la red

C.3.7.3. – Envío de mensajes (SMS)

Este ejemplo muestra como enviar un mensaje. Hay que poner el número de teléfono de destino. Para realizar mejor las medidas, se ha dejado las funciones de encendido y de conexión en el setup.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

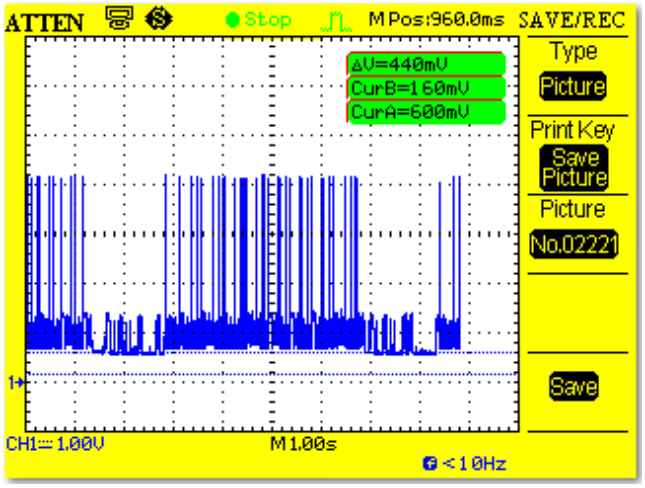


Fig. C.164.a. Medida general del proceso

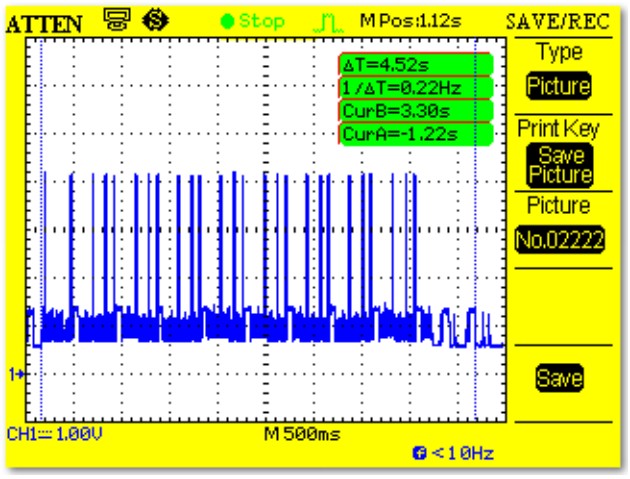


Fig. C.164.b. Tiempo de envio del sms

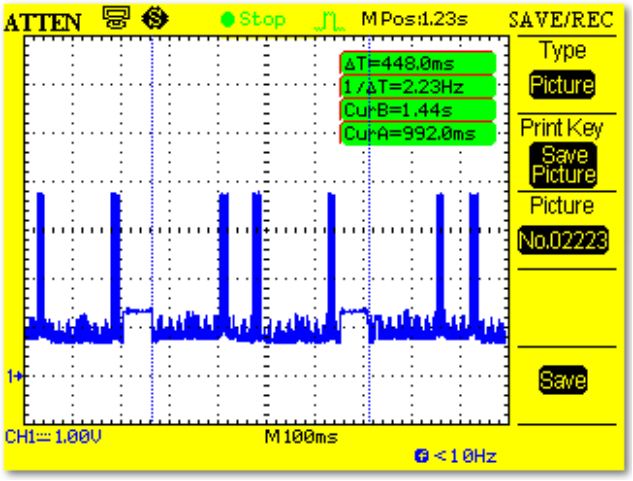


Fig. C.164.c. Tiempo de ejecución de una parte del proceso

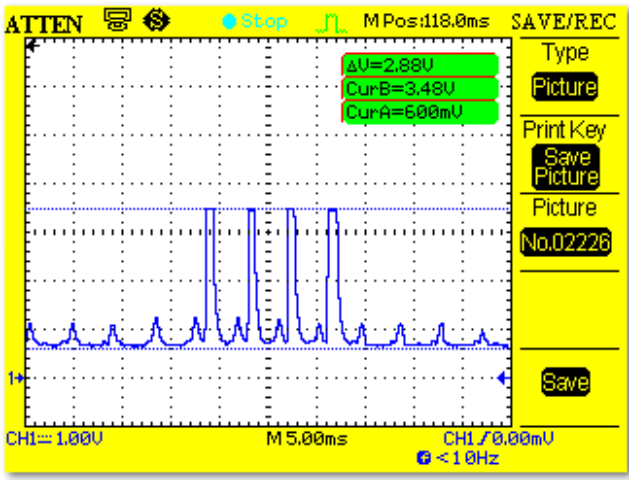


Fig. C.164.d. Consumo pulsos

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	4520	36.86	166608

Tabla C.171. Consumo del proceso del envío de SMS

C.3.7.4. – Lectura de una URL con petición GET

Este ejemplo muestra cómo leer una URL con una petición GET enviándose tramas a Meshlium usando el método GET. Libelium creó un servidor web para ello, configurándose la URL y el puerto. Con respecto al tamaño de las tramas, se pondrá un valor cercano al máximo, es decir, 130 bytes, añadiendo los bytes necesarios para tal fin.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

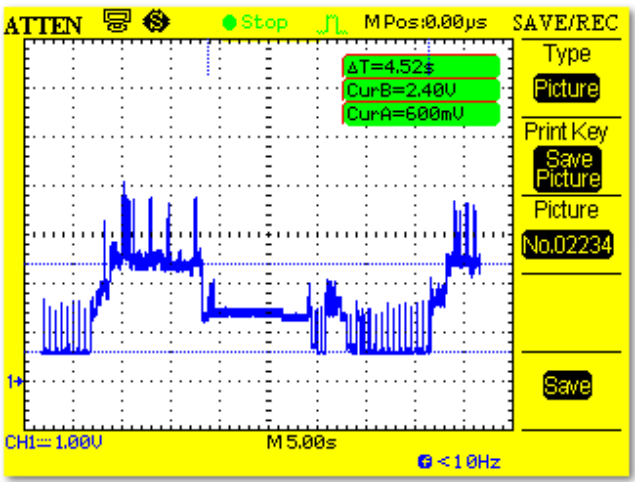


Fig. C.165.a. Medida general del proceso

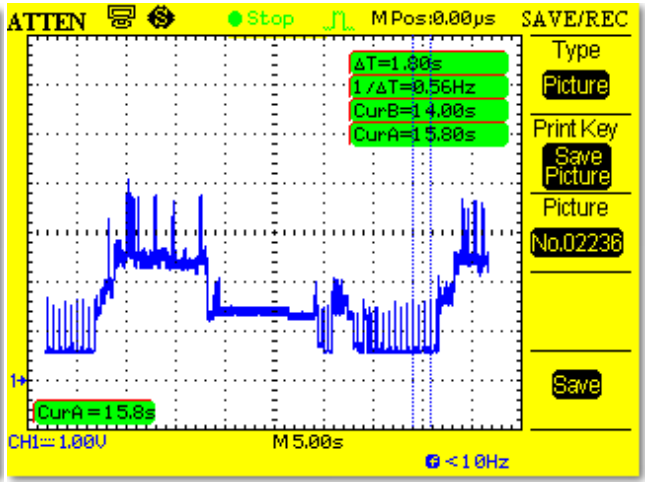


Fig. C.165.b. Tiempo de ejecución de las tramas

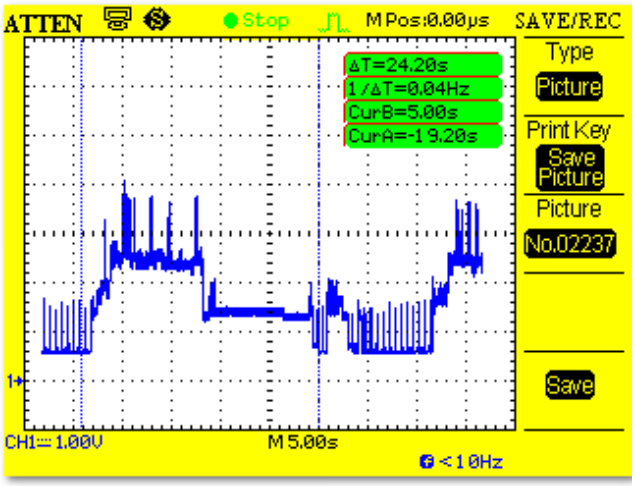


Fig. C.165.c. Tiempo de ejecución del proceso

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	26004	106.47	2768880

Tabla C.172. Consumo del proceso de la lectura de una URL enviándose tramas a Meshlium

C.3.7.5.- Envío de archivos al servidor FTP desde el módulo 3G

En este ejemplo se muestra como subir un archivo a un servidor FTP desde la tarjeta micro-SD del módulo 3G. Para ello, dejamos previamente encendido el 3G y que esté conectado a la red. Se seleccionará la carpeta y el nombre del archivo que quieres subir.

Es necesario configurar una serie de parámetros para tal fin. Se deberá de poner el correspondiente nombre de usuario, contraseña, servidor IP y el puerto del servidor FTP.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

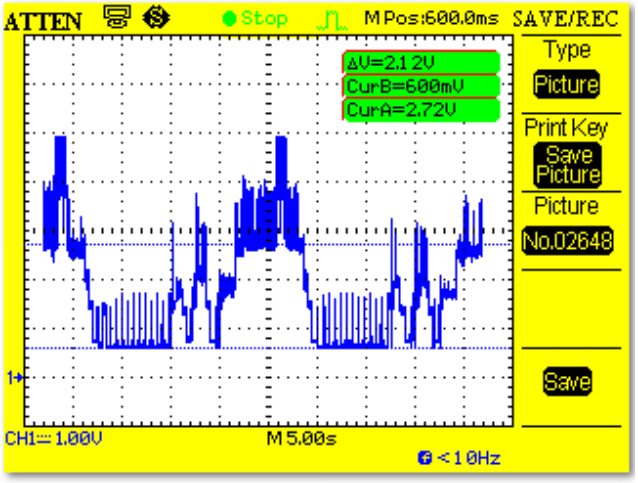


Fig. C.166.a. Medida general del proceso

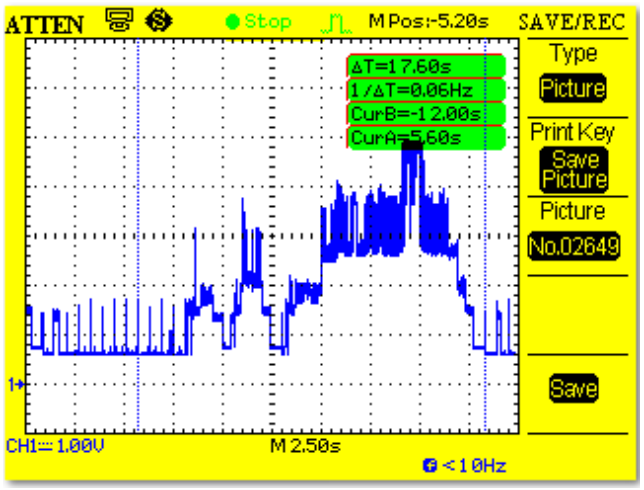
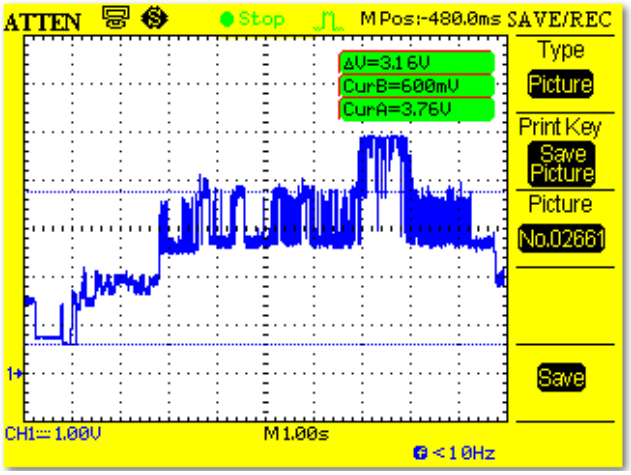


Fig C.166.b. Tiempo total del proceso completo



FigC.166.c. Proceso durante la subida del archivo

En la siguiente tabla se muestra el consumo de todo el proceso completo:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso Total	17600	154.44	2718300

Tabla C.173. Consumo proceso total del envio de archivos al servidor FTP desde el módulo 3G

C.3.7.6.- Envío de emails con SMTP

Este ejemplo muestra como enviar un email con SMTP. Para ello hace falta configurar una serie de parámetros del servidor SMTP como el nombre del servidor, el puerto al que está conectado, la cuenta y la contraseña de la cuenta. También, es necesario poner los parámetros del email como la cuenta de destino, el tema y el cuerpo del email.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

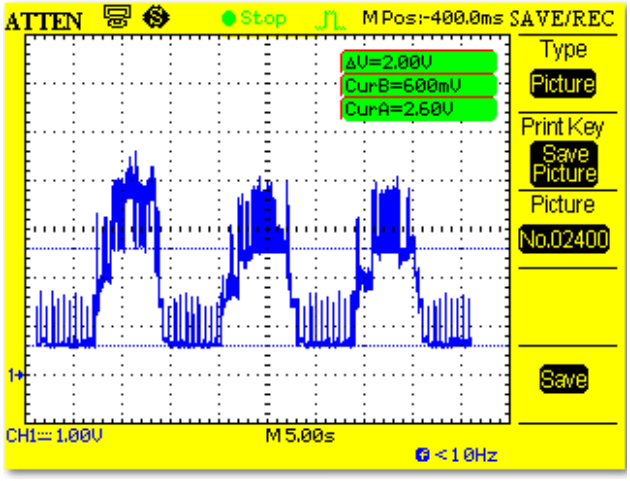


Fig. C.167.a. Medida general del proceso

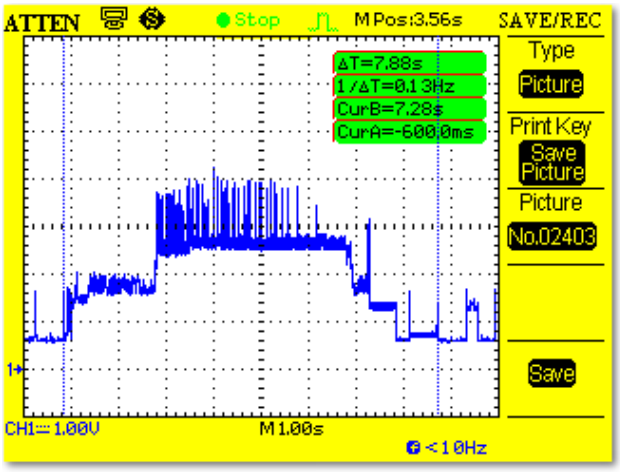


Fig C.167.b. Tiempo total del proceso completo

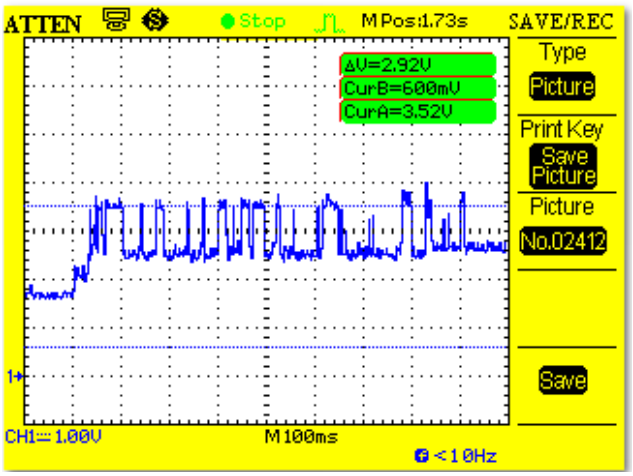


Fig C.167.c. Picos durante el proceso de envio

En la siguiente tabla se muestra el consumo de todo el proceso completo:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso Total	7880	147.96	116600

Tabla C.174. Consumo proceso total del envío de un email

C.3.7.7.- Envío de datos a través de TCP

Este ejemplo muestra cómo crear un cliente TCP no transparente en una sola conexión. Se realizará el envío de datos usando tramas.

Para ello, habrá que configurar el servidor TCP, poniendo la correspondiente dirección IP y el puerto.

Con respecto al tamaño de las tramas, se pondrá un valor cercano al máximo, es decir, 130

bytes, añadiendo los bytes necesarios para tal fin.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

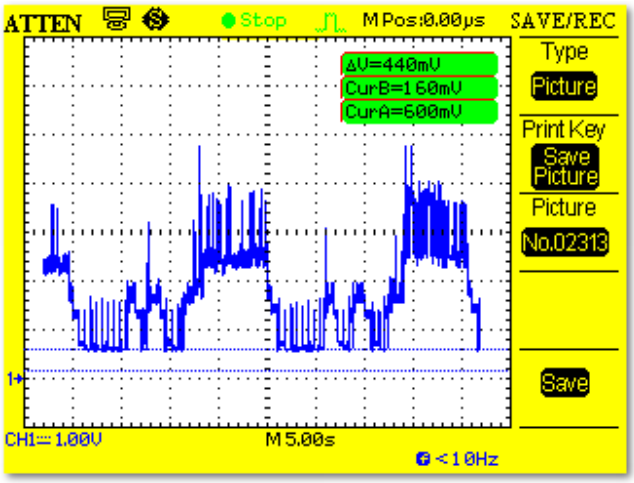


Fig. C.168.a. Medida general del proceso

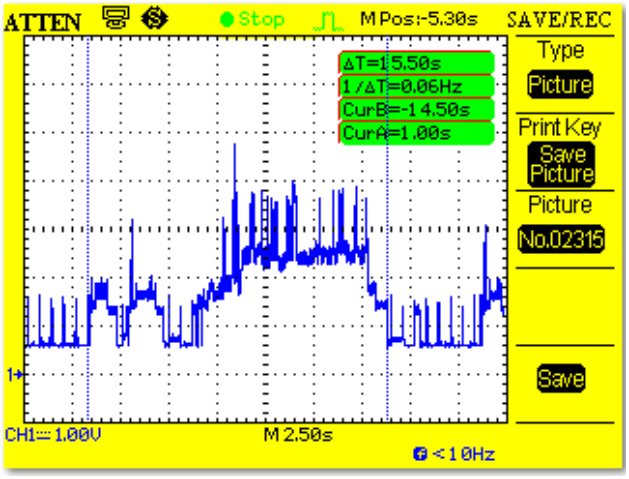


Fig. C.168.b. Tiempo de ejecución del proceso

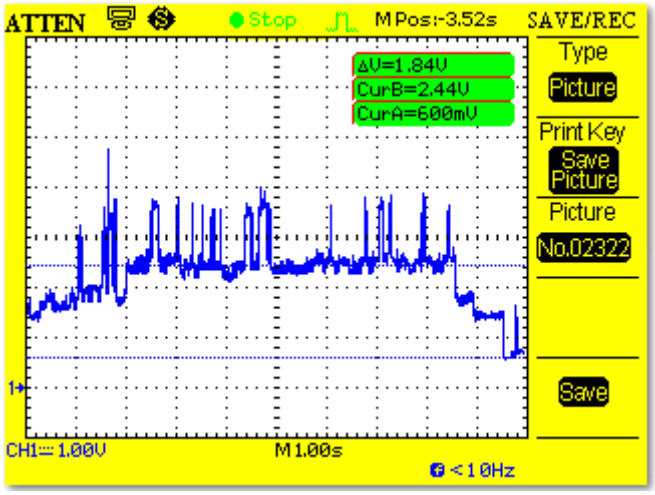


Fig. C.168.c. Proceso de envío de datos a través de TCP

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	15500	125.5	1945400

Tabla C.175. Consumo del proceso total para establecer conexiones TCP

C.3.7.8.- Envío de datos a través de UDP

Este ejemplo muestra cómo crear un cliente UDP no transparente en una sola conexión. Se realizará el envío de datos usando tramas.

Para ello, habrá que configurar el servidor UDP, poniendo la correspondiente dirección IP y el puerto.

Con respecto al tamaño de las tramas, se pondrá un valor cercano al máximo, es decir, 130 bytes, añadiendo los bytes necesarios para tal fin.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

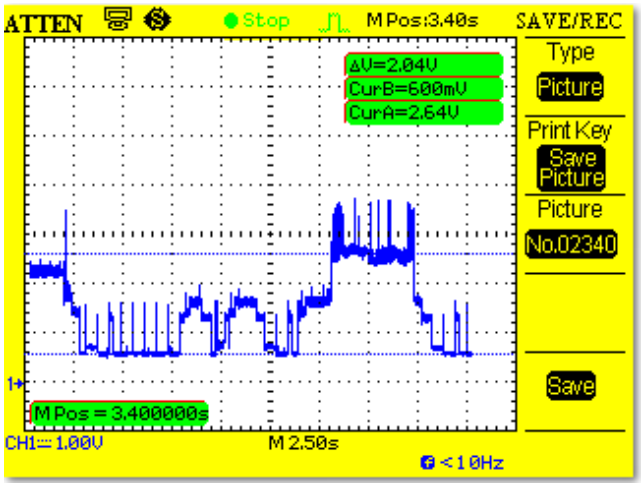


Fig. C.169.a. Medida general del proceso

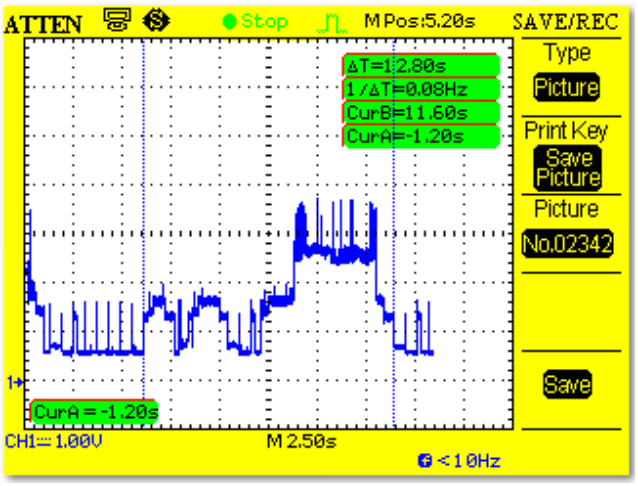


Fig. C.169.b. Tiempo de ejecución del proceso

El consumo del proceso total es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso total	12800	114.9	1470840

Tabla C.176. Consumo del proceso total para establecer conexiones UDP

C.3.7.9.- Modos GPS

C.3.7.9.1. – Proceso de encendido y apagado del GPS

Este ejemplo muestra como encender y apagar el receptor GPS del módulo 3G.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

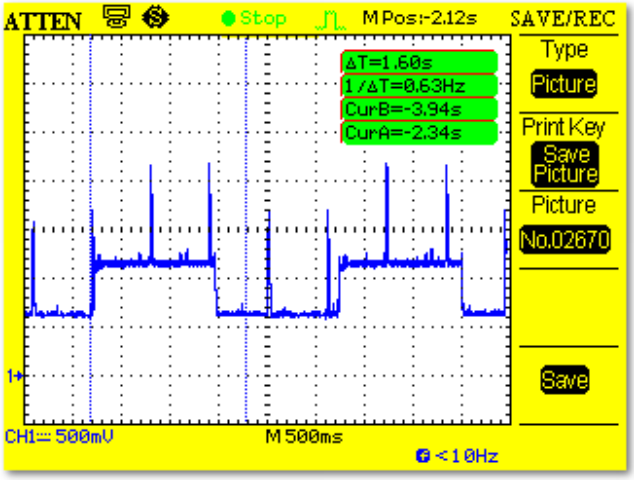


Fig. C.170.a. Proceso total

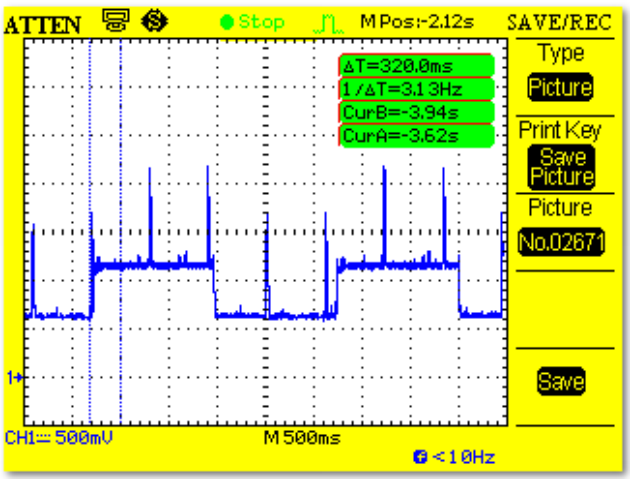


Fig. C.170.b. Proceso on

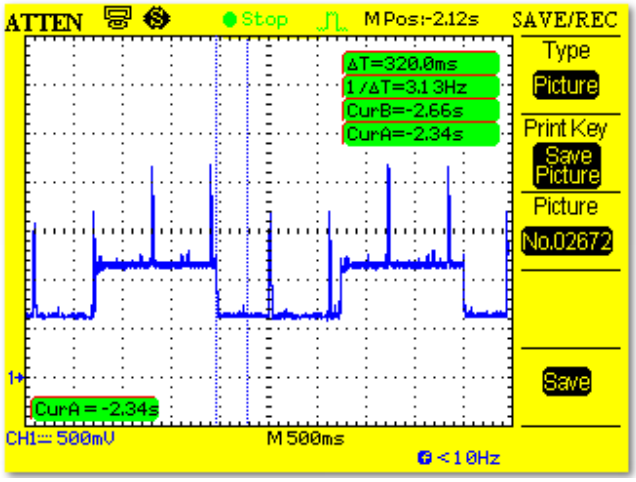


Fig. C.170.c. Proceso off

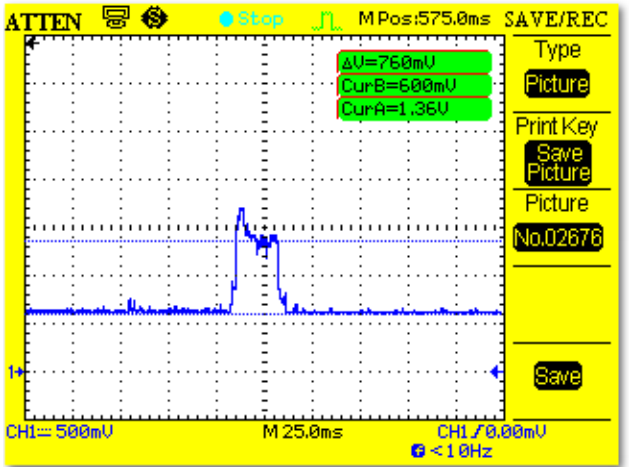


Fig. C.170.d. Consumo de los picos

El consumo en el proceso y estado on y en el proceso off del receptor GPS es el siguiente:

	Duración Total (ms)	Corriente total (mA)	Carga Energética total (mA*ms)
Proceso On	320	57.56	18420
Estado On	1000	57	57000
Proceso Off	320	2	640

Tabla C.177. Consumo del proceso y estado on y del proceso off del receptor GPS

C.3.7.9.2. – Modo Stand Alone

Este ejemplo muestra cómo iniciar el GPS en modo autónomo y espera a que los datos GPS estén disponibles. Cuando los datos GPS se encuentran disponibles lo muestra que cada dos segundos.

Como las pruebas se realizaron al aire libre, en el código, se añadieron una serie de funciones para saber con seguridad si el módulo se conecta o no. Si el led 0 se enciende y se apaga es porque se ha llegado a conectar el módulo y si el Led 1 permanece todo el rato encendiéndose y apagándose es porque no hay conexión. Normalmente el módulo se conecta en menos de un minuto. Si tarda más de 240 segundos, que es el rango límite de conexión, es que no se llega a conectar. En algunas ocasiones hay que darle tiempo a que se conecte. No siempre pasa en menos de un minuto.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

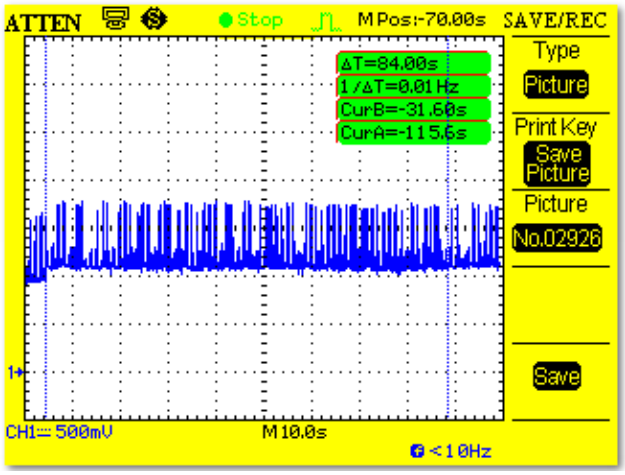


Fig. C.171.a. Tiempo de conexión

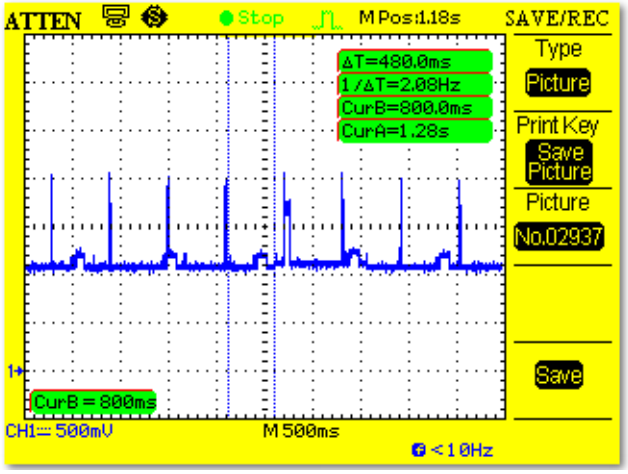


Fig. C.171.b. Tiempo proceso lectura información del GPS

Como se ve en la gráfica (a), el tiempo de espera para la conexión del GPS es de 84 segundos.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	84480	5.66	478720

Tabla C.178. Consumo del proceso total

C.3.7.9.3. – Modo MS-based

Este ejemplo muestra cómo empezar el GPS en modo Ms-based y espera a que los datos GPS estén disponibles. Cuando estos lo están, se muestran cada dos segundos.

Como las pruebas se realizaron al aire libre, en el código, se añadieron una serie de

funciones para saber con seguridad si el módulo se conecta o no. Si el led 0 se enciende y se apaga es porque se ha llegado a conectar el módulo y si el Led 1 permanece todo el rato encendiéndose y apagándose es porque no hay conexión. Normalmente el módulo se conecta en menos de un minuto. Si tarda más de 240 segundos, que es el rango límite de conexión, es que no se llega a conectar. En algunas ocasiones hay que darle tiempo a que se conecte. No siempre pasa en menos de un minuto.

A continuación, se muestran las gráficas como resultado del ejemplo ejecutado:

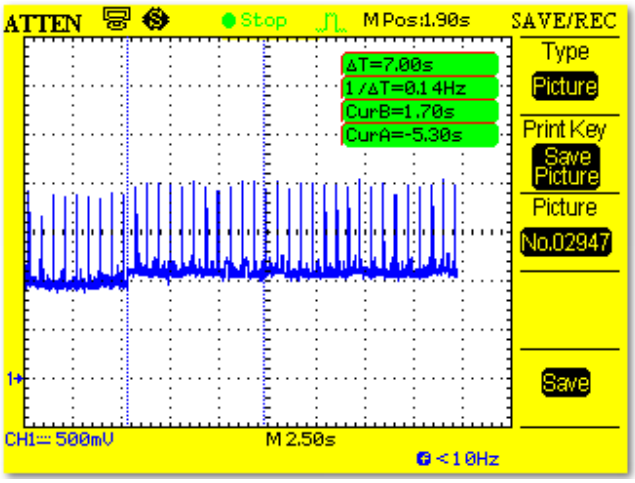


Fig. C.172.a. Tiempo de conexión

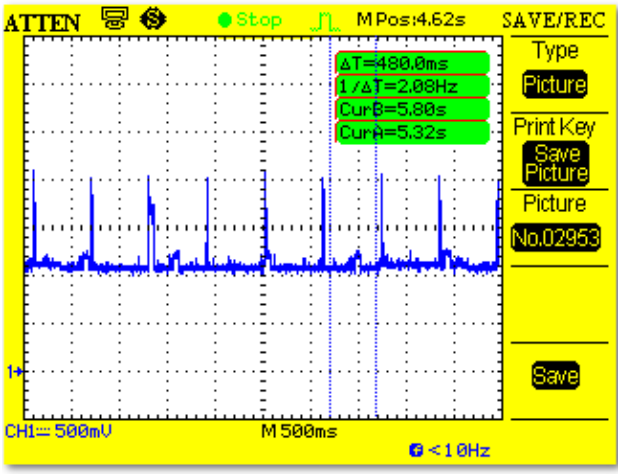


Fig. C.172.b. Tiempo proceso lectura información del GPS

Como se ve en la gráfica (a), el tiempo de espera para la conexión del GPS esta vez es menor que la otra vez. Le cuesta conectarse 7 segundos.

El consumo del proceso total es el siguiente:

	Duración Total (ms)	**Corriente total (mA)**	**Carga Energética total (mA*ms)**
Proceso total	7480	4.915	36768.8

Tabla C.179. Consumo del proceso total

Anexo D: API del Waspmote.

El API del Waspmote se divide en dos carpetas: "*cores*" y "*Libreries*". Las librerías dentro de "núcleo" se invocan automáticamente lo que no hay necesidad de añadirlos a la “. *Pde*". Algunos ejemplos son "*Utils.h*" o "*WaspACC.h*". Sin embargo, es obligatorio incluir manualmente a la ". pde", una librería que se encuentra dentro de la carpeta "libreries" (si es que tenemos que usarlo).

Las librerías utilizadas en los programas deben estar incluidas al principio del código escrito por la inclusión de la cabecera correspondiente.

La lista de clases que se incluyen cuando se utilizan son:

```
#include <WaspBT_Pro.h>
#include <WaspBLE.h>
#include <WaspFrame.h>
#include <WaspGPRS_Pro.h>
#include <WaspGPS.h>
#include <WaspRFID1356.h>
#include <WaspSensorAgr_v20.h>
#include <WaspSensorAmbient.h>
#include <WaspSensorCities.h>
#include <WaspSensorEvent_v20.h>
#include <WaspSensorGas_v20.h>
#include <WaspSensorParking.h>
#include <WaspSensorPrototyping_v20.h>
#include <WaspSensorRadiation.h>
#include <WaspSensorSW.h>
#include <WaspSensorSmart_v20.h>
#include <WaspWIFI.h>
#include <WaspXBee802.h>
#include <WaspXBee868.h>
#include <WaspXBee900.h>
#include <WaspXBeeZB.h>
#include <WaspXBeeDM.h>
#include <Wasp3G.h>
```

El código también está formado por una serie de variables globales que se implementan antes de la función setup del código. Por ejemplo:

```
// Global variables declaration
```

```
char* str = ”This is a string”;
uint8_t number = 0;
int counter = 0;
void setup()
{
}
void loop()
{
counter++;
USB.println(str);
number = 121;
}
```

Anexo E: Waspmote IDE [13].

Compilador clave para la ejecución de los códigos y utilizado en todo momento. Es un entorno de desarrollo integrado para Waspmote. Se utiliza para escribir el código y subirlo a Waspmote. También se usa para monitorizar las salidas serie y para la depuración. Este IDE contiene la API de Waspmote (la API es el conjunto de todas las bibliotecas de Waspmote)

Van saliendo nuevas versiones de API instantáneamente por Libelium siempre que se hagan mejoras o arreglos de bugs.

Hay varias ventajas para utilizar este nuevo Waspmote IDE:

- Compilación más rápido
- Nueva estructura de la API
- Más mensajes de depuración
- Información sobre la memoria RAM utilizada
- Una instalación más fácil
- Más preferencias
- Más de 30 idiomas diferentes
- Nuevas actualizaciones automáticas
- compatibilidad OTA
- Pestañas editor desplazables

El usuario debe tener un IDE para cada API. De acuerdo con la nueva organización de las librerías para Waspmote v1.2, hay librerías del núcleo que son únicos para cada IDE. Además, hay una carpeta adicional con las librerías opcionales. Así que debido a las librerías del núcleo único, si el usuario quiere experimentar con 'n' APIs, necesita tener 'n' IDEs instalados en el ordenador. La nueva organización de librerías reduce el uso de memoria RAM, y mantiene todo Como programa portable, se puede tener tantos IDEs como quieras.

La ubicación de la API para cada sistema operativo es:

Windows:

- Examples: <Waspmote IDE folder>\examples\
- Libraries: <Waspmote IDE folder>\libraries\
- API core: <Waspmote IDE foler>\hardware\waspmote\cores\waspmote-api\

Las principales características del Waspmote IDE son:

Writing Sketches:

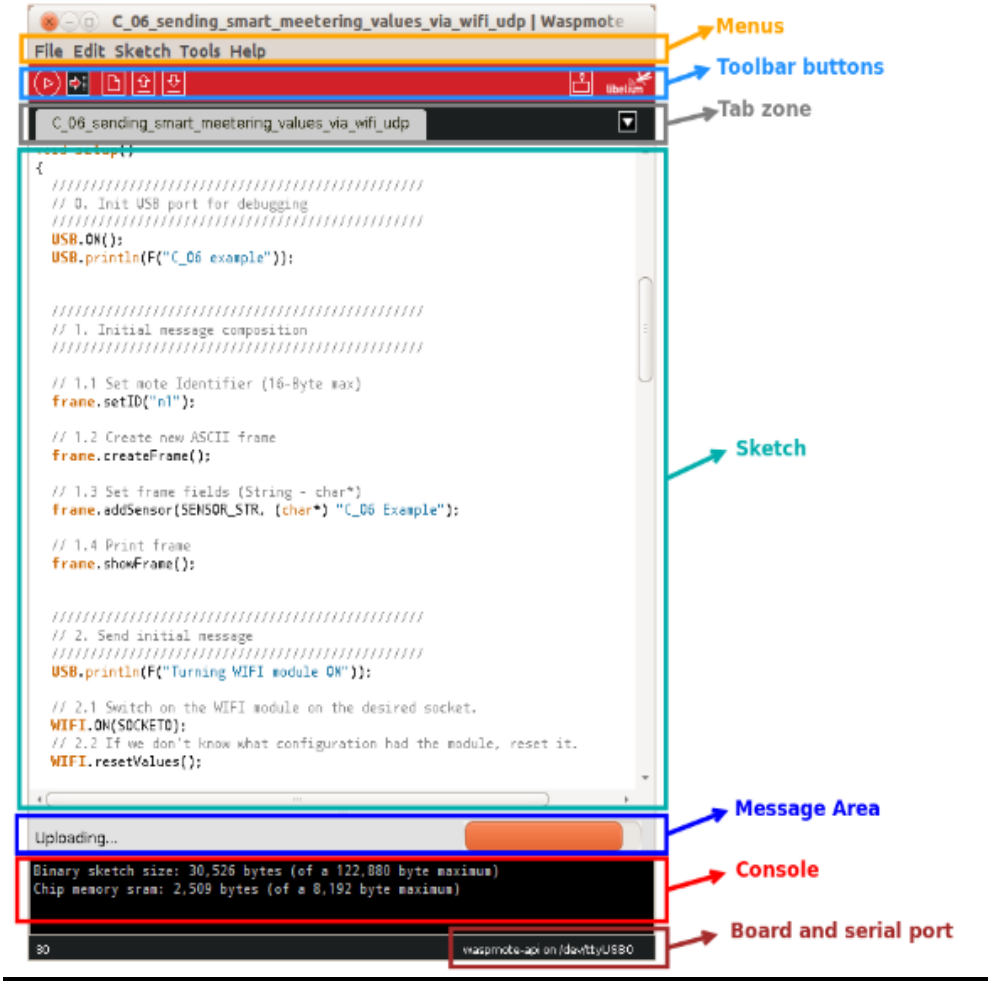


Fig. E.1. Las diferentes secciones que forma el Waspmote IDE

El software escrito utilizado en el IDE se llama "Sketches". Estos se escriben en el editor de texto. Se guardan en un archivo cuya extensión es ". PDE". El área de mensajes proporciona información mientras se guarda y se produce, además, la exportación de errores. La consola muestra la salida de texto por el IDE incluyendo los mensajes de error completos y otra información. La esquina derecha inferior de la ventana muestra la placa y el puerto serie al que está conectado.

Este IDE permite administrar "sketches" con más de un archivo (en la zona sub-menú). Pueden ser archivos normales de código (Waspmote. PDE), archivos de C (. Extensión c), archivos de C + + (. Cpp), o archivos de cabecera (. H).

Menus:

Los botones de la barra de herramientas permiten comprobar y cargar programas, crear, abrir y guardar "*sketches*", y abrir el "serial monitor".

- Botón de compilar : Comprueba el código de error.
- Botón de subir : Compilar el código y lo sube a la placa Waspmote.
- Botón de nuevo : Crea un nuevo dibujo.

- Botón de abrir : Presenta un menú con todos los bocetos en su cuaderno de bocetos. Al hacer clic en uno se abrirá en la ventana actual.
- Botón de guardar : Guarda tu dibujo.
- Botón de serial Monitor : Abre el monitor serie.

Los comandos adicionales se encuentran dentro de los cinco menús: Archivo, Edición, Sketch, Herramientas, Ayuda.

Uploading:

Antes de subir el “sketch” correspondiente a compilar, es necesario seleccionar los elementos correctos de la instrumentación.

Tenemos que seleccionar el puerto serie correcto, dentro de “Herramientas”, y una vez hecho esto, se presiona el botón de subir en la barra de herramientas o seleccionar el elemento Upload del menú Archivo. Waspmote se restablecerá automáticamente y comenzará la carga. Se mostrará un mensaje cuando a carga esté completa o un error, en caso que ocurra.

Anexo F: Guía Energética para el usuario.

La Guía energética (se encontrará en el DVD) es un documento, redactado en inglés, donde se plasman todos los consumos eléctricos de los distintos modos y operaciones de funcionamiento de la plataforma Waspmote, incluyendo sus sensores y módulos de comunicaciones asociados a través de medidas experimentales.

Esta guía le sirve de utilidad al usuario para que sepa de manera rápida lo que consume cada acción que puede realizar la plataforma Waspmote y dependiendo de las seleccionadas, el usuario puede calcular de manera rápida y sencilla la vida de la batería del dispositivo.

Las guías energéticas, son cada vez de mayor relevancia en productos electrónicos de todo tipo y especialmente, en redes de sensores inalámbricas.

El documento está diferenciado en 3 partes principales:

- Funcionalidades “internas” del Waspmote.
- Placas de sensores integrados.
- Módulos de comunicaciones.

Cada hoja deberá tener:

- El título de la funcionalidad que se mide.
- Un link que irá al código ejecutado para realizar la medición de la prueba.
- Una serie de pantallazos del osciloscopio correspondientes a las medidas realizadas.
- Una tabla donde se vean 3 datos del consumo: la anchura total del proceso (en ms), la altura media (en mA) y el cálculo de la carga energética (es la multiplicación de los 2 anteriores [en mA*ms]).
- Al final habrá, cuando resulte necesario, una serie de comentarios o notas. Especialmente importante cuando consume mucho, es lento, pasan cosas raras, consejos de uso que le da al usuario o si no se ha seguido los criterios estándar.

Cabe destacar que los links que se han puesto en cada ejemplo, no corresponden a los reales ya que serán los propios compañeros de la empresa los que se encarguen de poner el link correcto para subirlos a la web. Solo están puesto de guía para saber mis compañeros que ruta seguir para acceder a ellos.

Respecto a las gráficas, se pone el nombre de cada imagen, y serán los compañeros del departamento de diseño, los que se encarguen de ponerlos como ellos crean conveniente.