ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Proyecto Fin de Carrera

Sistema de monitorización ambiental basado en hardware open-source alimentado mediante energía fotovoltaica (Environmental monitorization system based on open-source hardware powered by photovoltaics)

Para acceder al Título de

INGENIERO TÉCNICO DE TELECOMUNICACIÓN

Autor: Óscar López Ruiz

Julio - 2014

E.T.S. DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACION

INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN

CALIFICACIÓN DEL PROYECTO FIN DE CARRERA

Realizado por: Oscar López Ruiz
Director del PFC: Jesús Mª Mirapeix Serrano

Título: "Sistema de monitorización ambiental basado en hardware open-source alimentado mediante energía fotovoltaica"
Title: "Environmental monitorization system based on open-source hardware powered by photovoltaics"

Presentado a examen el día: 17 de Julio de 2014

para acceder al Título de

INGENIERO TÉCNICO DE TELECOMUNICACIÓN, ESPECIALIDAD EN SISTEMAS ELECTRÓNICOS

Composición del Tribunal:
Presidente (Apellidos, Nombre): Cobo García, Adolfo
Secretario (Apellidos, Nombre): Mirapeix Serrano, Jesús
Vocal (Apellidos, Nombre): Arce Hernando, Jesús Antonio

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente

Fdo.: El Secretario

Fdo.: El Vocal

Fdo.: El Director del PFC
(sólo si es distinto del Secretario)

Vº Bº del Subdirector

Proyecto Fin de Carrera Nº
(a asignar por Secretaría)

Agradecimientos:

A mi familia y amigos por apoyarme durante todo este tiempo. Nunca se dirá lo suficiente pero gracias, de verdad.
A Jesús por tener siempre tiempo para atenderme y resolver mis dudas.

Siempre permanecerán los buenos recuerdos, con la esperanza de volver a repetir tan grandes momentos.

Gracias a todos.

ÍNDICE

1. Introducción

1.1. Contexto

1.1.1. Sistemas de monitorización

A día de hoy, la **monitorización** resulta imprescindible para poder gestionar sistemas (de todo tipo) de una forma **segura** y **eficiente**. Para poder llevar a cabo esta misión, existe una amplia gama de sensores en el mercado dependiendo de la magnitud a medir y de las prestaciones que se requieran. Así, es posible encontrar sensores de: **temperatura**, **humedad**, **presión**, **fuerza**, etc.

El estudio de las condiciones ambientales se lleva practicando desde mucho tiempo atrás. Ya en la antigüedad existían métodos para predecir buenas estaciones de cosecha, festividades, etc. No es sino la condición humana la que incita a conocer nuestro **entorno** para, por una parte, **adaptarnos** al medio y, por otra, poder aprovechar los recursos que éste nos ofrece. Conocer lo que nos rodea nos ayuda a planificar tareas, comportamientos... Todo esto supone la supervivencia y la mejora en la calidad de vida de una sociedad y de una especie.

Con la aparición de equipos de **bajo coste** que pueden llevar a cabo esta tarea de manera sencilla y sin muchos recursos, es necesario conocer sus aplicaciones, **prestaciones** y **limitaciones** que pueden tener para elegir adecuadamente cuál es apto y cuál no para cierto sistema.

Adicionalmente, la inclusión de este tipo de sensores en un **sistema aislado** nos da la posibilidad de poder monitorizarlo de forma remota, sin tener acceso físico a él y, con ello, reducir costes de inspección y control. Sin embargo, la implementación de estos sistemas puede llegar a ser **tediosa**. Dependiendo del número de elementos a monitorizar, las magnitudes a medir y factores externos, la **complejidad** del proyecto puede aumentar sustancialmente. Por ello, existen multitud de herramientas de análisis y caracterización que ayudarán al técnico encargado del sistema a cumplir los requerimientos exigidos y obtener los resultados adecuados.

La llegada de soluciones "***hardware***" asequibles al mercado de masas ha hecho posible montar, **automatizar** y configurar todo un sistema bajo la dirección de una única plataforma, de forma sencilla y **eficiente**. La solución que se plantea en este documento va encaminada justamente en esa dirección.

Gracias al gran soporte de este tipo de plataformas por parte de los **usuarios**, es posible encontrar todo tipo de proyectos en la **Red**; desde los más sencillos a los que más complejidad entrañan. Es, gracias a esta comunidad, que los dispositivos de *"hardware"* libre están de moda en estos días, facilitando enormemente el trabajo en **aplicaciones rutinarias**, procesos automatizados, monitorización, etcétera.

Otra parte importante del presente proyecto es la comunicación entre los distintos componentes. En este caso, se realiza por medio de **radiofrecuencia**, una tecnología que ofrece la posibilidad de implementar un sistema **sin cables**, de **largo alcance** y **seguro**, permitiendo así **independencia** del conjunto de medios físicos, como pueden ser cables y otros elementos que dificulten la instalación. Este tipo de transmisión, además, es óptima para el

envío de información **cíclica** debido a su gran eficiencia a la hora de transmitir: se puede llegar a conseguir autonomías de **meses**, incluso **años** con una simple pila o batería.

Y, por otra parte, la elección de un sistema de alimentación por medio de energía **fotovoltaica** refuerza esa independencia comentada anteriormente, al no depender de una red eléctrica cercana, permitiendo la instalación en **entornos remotos**.

1.1.2. Soluciones comerciales

Con la aparición de soluciones basadas en ***Arduino*** [1] y ***Raspberry Pi*** [2] es posible desarrollar un proyecto de **sistemas embebidos**, tan demandado en estos momentos, con poco desembolso. Estos dispositivos ofrecen una solución eficaz para tareas sencillas e incluso algunas más complejas, todo depende de los **módulos** utilizados y la programación a desarrollar. Las herramientas necesarias para llevar a cabo proyectos de este tipo están al alcance de **todos** y requieren **poca curva de aprendizaje,** ya que están diseñados para usuarios tanto noveles como profesionales. Además, son dispositivos ya asentados y usados por **gran cantidad de usuarios**, por lo que, por un lado ofrecen una gran ventaja ya que se dispone de **ejemplos** y **guías** que ayudan a adquirir conocimientos necesarios aunque, por el contrario, resulta difícil ser realmente **innovador** en este campo.

Desde su creación y puesta en producción, estas dos plataformas (en concreto) han tenido una gran **acogida**. ***Arduino*** [1] porque facilitó el desarrollo de microcontroladores en entornos universitarios y enseñanza en general a un bajo coste y ***Raspberry Pi*** [2] porque desde sus inicios estaba diseñada para alentar a los más jóvenes a empezar a programar.

Respecto al apartado **económico**, este tipo de soluciones de licencia libre permiten al usuario desarrollar un equipo completo de monitorización, por ejemplo, de una manera fácil y **asequible**; mientras que de otra forma sería imprescindible recurrir a tecnologías de ámbito industrial, **aumentando** con ello el precio del producto final. Por ello, la elección de los distintos sensores y placas se ha basado principalmente en **dos aspectos**: la relación **calidad/precio** y el aprovechamiento de **ofertas** puntuales en los distintos componentes; queriendo así transmitir al lector una reflexión sobre la necesidad de invertir tiempo en buscar la solución **óptima** a un problema concreto.

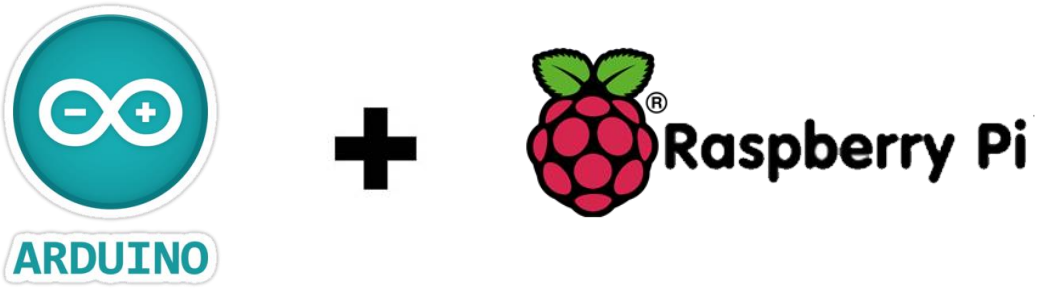


Figura 1. Asociación entre plataformas Arduino y Raspberry Pi.

1.1.3. Componente social

Por otro lado, la **plataforma *web*** utilizada está basada en lenguajes de programación y estructuras de código abierto, así que son fáciles de mantener y están en continuo desarrollo, agregando nuevas características cada cierto tiempo contrastadas por una gran cantidad de usuarios. Además, existen gran cantidad de **librerías** y **funciones** que no sólo ayudan a aquel que da sus primeros pasos en este tipo de tecnologías, sino que son usadas por la mayoría de proyectos similares.

Para comprender la cantidad de posibilidades que tienen estas plataformas sólo hace falta ver el volumen de mensajes, entradas en *blogs*, artículos en prensa, etc. que generan. Se ha abierto un nuevo paradigma alrededor de estas plataformas, dando libertad en cuanto hardware antes no existente; primando sobre todo la filosofía ***"do it yourself"*** donde cada uno intenta, con sus medios, formar el sistema por sí mismo. En la Red podemos encontrar gran cantidad de ejemplos distribuidos por los propios usuarios para que, en caso de necesidad o propia curiosidad, se tenga una guía o primeros pasos por dónde empezar. Además, existen blogs, foros, etcétera, con recopilaciones de algunos de estos proyectos donde simultáneamente se forma una conexión entre el creador del proyecto y distintos usuarios que preguntan, se informan y retroalimentan entre ellos mismos. Así es como se forman las comunidades.

Como contrapunto, como se ha comentado anteriormente, debido a la gran cantidad de usuarios que utilizan este tipo de soluciones resulta muy difícil crear innovador. Sin embargo, tomando un ejemplo como base se puede personalizar a las necesidades de cada usuario. Estos es importante ya que un mismo proyecto base puede derivar en varios con funcionalidades completamente diferentes.

1.2. Motivación

La motivación de este proyecto es **personal**.

Una de las principales motivaciones de este proyecto es la de plantear un sistema de monitorización completo y robusto de forma barata. Un sistema autónomo capaz de recoger los datos ambientales de una instalación remota, o bien en el propio hogar.

La intención de este proyecto es la de ofrecer una alternativa a otras soluciones del mercado con el añadido de ser asequible y tremendamente configurable por el usuario. Por ello, a lo largo de este documento se intentará desarrollar la configuración de este sistema desde la vista de un usuario medio.

A través de las diferentes secciones se intentará aportar al lector una idea general del proyecto y, como información adicional, la puesta en marcha del sistema al completo junto a las características técnicas del mismo.

1.3. Objetivos

El objetivo de este proyecto es construir un equipo de **monitorización** aplicado a un sistema fotovoltaico, **aislado**, para controlar en **tiempo real** las condiciones que lo rodean y actuar en consecuencia.

Como principal atractivo está el **bajo coste** del sistema y la simplicidad de configuración del mismo. A esto añadimos el concepto de "***open-source***", es decir, código abierto, tanto "hardware" como "software". El sistema se puede **adaptar** a las necesidades requeridas cambiando libremente el código proporcionado y, con ello, personalizar todo el entorno del proyecto para dotarlo del valor añadido de haber formado un sistema desde los cimientos.

Las plataformas usadas en este proyecto cuentan con una gran **comunidad** en la Red que puede resolver cualquier duda que pueda surgir. Existen, asimismo, gran cantidad de **ejemplos** y **manuales** de uso de todas las herramientas necesarias.

La idea principal de este proyecto, como se ha comentado anteriormente, es ofrecer una herramienta de monitorización aplicada a un sistema fotovoltaico. Sin embargo, el sistema puede ser usado en multitud de **aplicaciones**, como por ejemplo: tanques de **contención** de **líquidos**, **alimentos**, etcétera; conocer los **datos ambientales** de lugares donde no se pueda tener acceso físico a los datos de un sensor, y cualquiera otra aplicación que requiera un bajo presupuesto y no se disponga de línea de alimentación tradicional.

Como **meta** final de este proyecto se ha planteado el diseñar un sistema que sea sencillo de utilizar y fácil de mantener. Con unos simples ajustes se puede configurar el **tiempo de muestreo**, la información mostrada, **diseñar gráficos** con los datos recogidos, y un largo etcétera.

A esto hay que añadir también un componente **social,** ya que desde la plataforma *web* se podrán compartir los datos de consumo, temperatura y humedad a través de redes sociales.

Un **resumen** de los objetivos a lograr con el presente proyecto son:

- Sistema de monitorización sencillo de instalar y fácil de mantener.
- Sistema económico.
- Sistema seguro y de gran alcance.
- Sistema totalmente configurable por el usuario adaptándose a sus necesidades.
- Sistema independiente de medios físicos de alimentación.
- Sistema autónomo para instalaciones en lugares remotos.
- Sistema con componente social para atraer a otros usuarios a la plataforma.

1.4. Estructura del documento

- **Introducción**: Breve descripción del proyecto, contexto y motivación. Objetivos y organización del documento.
- **Elementos del sistema**: Una aproximación a los elementos que conforman el sistema. Conjunto y esquema.
- ***Raspberry Pi***: Qué es, para qué sirve y cuál es su papel en el proyecto. Ventajas y limitaciones.
- ***Arduino***: Pequeña introducción a la plataforma y a microprocesadores de bajo coste.
- **Alimentación**: Explicación de la forma de energía elegida y el sistema de acumulación utilizada.
- **Aplicaciones web**: Plataforma web usada para mostrar los datos, interfaz con los sensores y servicios adyacentes.
- **Lenguajes de programación**: Introducción de los lenguajes utilizados durante el proyecto.
- **Pruebas y resultados**: Pruebas con los sensores, datos de consumos y de autonomía.
- **Resumen, conclusiones y líneas futuras**: Posibles aplicaciones y desarrollo de nuevos sistemas.

2. Componentes del sistema

A continuación se explicará el funcionamiento del sistema de monitorización:

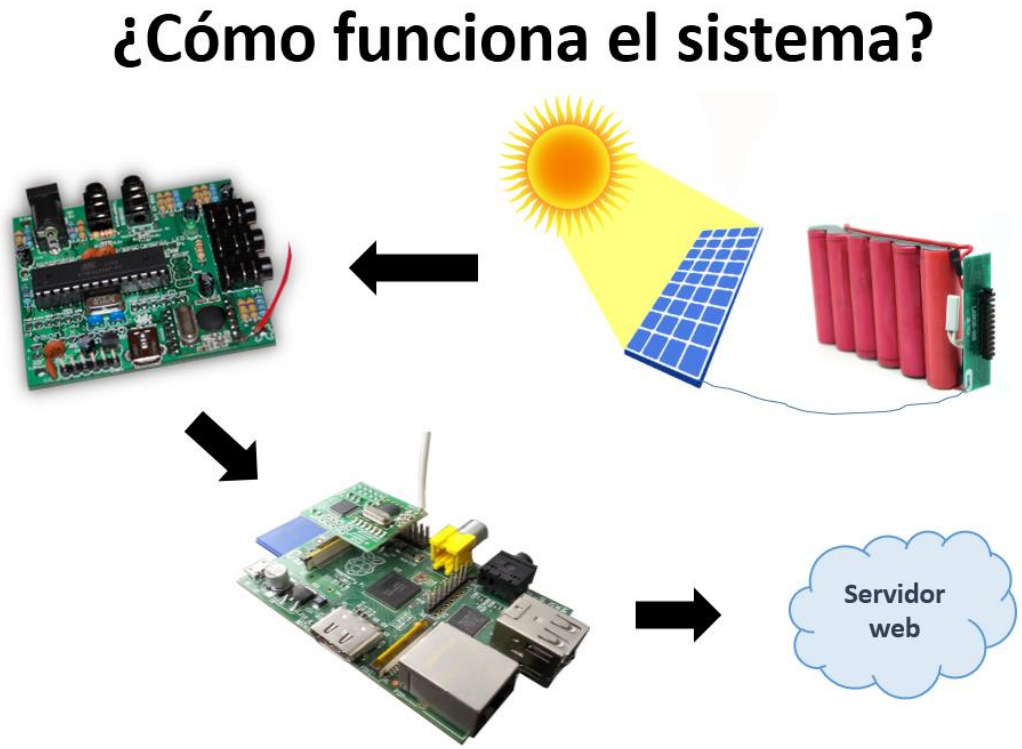


Figura 2. Esquema general del sistema de monitorización.

Como se observa en la figura 2, el sistema está formado por (de derecha a izquierda siguiendo las flechas): una placa solar y un conjunto de baterías encargados de alimentar el sistema, una placa basada en la plataforma ***Arduino*** [1] junto a los sensores, cuya función es recoger los datos ambientales que más tarde se transmitirán al siguiente elemento, en este caso se trata de una ***Raspberry Pi*** [2], ordenador de placa reducida o placa única (***SBC***). Su función es la de dar formato a los datos recibidos, almacenarlos en un base de datos y, si se dispone de conexión a Internet (mediante cable ***Ethernet*** o adaptador ***Wifi***), los almacenará en la nube en un servidor externo. Si no se dispone de conexión a Internet

Una vez alojados los datos, el usuario podrá de crear **gráficas**, consultar los **históricos**, etc. de toda esa información almacenada desde la propia plataforma web. Asimismo, se incluirán módulos especiales para **controlar** distintas **funciones** del sistema, para **representar** datos concretos, etc. Un ejemplo de estos módulos puede ser la opción de crear un **informe** con los datos recogido entre un **rango** de **fechas seleccionado**.

Más adelante se detallarán estas opciones adicionales junto a sus correspondientes implicaciones en el sistema.

2.1. Introducción a Arduino

El proyecto *Arduino* [1] surgió como un proyecto para estudiantes en el Instituto IVREA, en Ivrea (Italia). Poco a poco ha surgido una legión de seguidores de esta plataforma de *hardware* abierto, debido principalmente a su sencillez y a la multitud de tareas que puede desarrollar. Las placas Arduino utilizan un microcontrolador **Atmel AVR** encargado de dirigir los puertos de entrada/salida. Los microcontroladores de la familia **ATMEGA** son de los más usados dentro de la plataforma y, concretamente, se encargan de dirigir el flujo de datos en este proyecto.

El lenguaje de programación utilizado en este tipo de placas es el llamado Processing/Wiring. Su sintaxis es parecida a la de C y lenguajes similares. Permite crear un cargador de arranque o ***bootloader*** cuya misión es iniciar las funciones del microcontrolador. El entorno de desarrollo utilizado en estos casos es "**Arduino IDE**". Basado en java, ofrece un editor de texto y una consola de depuración junto a las opciones para cargar en el microcontrolador el *bootloader* desarrollado por medio de un cable USB-FTDI.

En este caso, el microcontrolador elegido es el **ATMEGA328**, un chip de bajo coste pero de prestaciones suficientes para el propósito del proyecto. Como la mayoría de placas Arduino, cuenta con un gran número de puertos de salida y entrada. Estos puertos servirán de pasarela para mandar información al microcontrolador. Una vez que la información es recibida por el "cerebro", éste la distribuye, formatea o la redirige hacia otros puertos o dispositivos.

Debido a la gran demanda de este tipo de placas, han surgido multitud de diseños, modelos, copias, hasta imitaciones. Al tratarse de dispositivos de *hardware* libre son fácilmente replicables aunque sólo reciban soporte los dispositivos 'oficiales'.

En las siguientes páginas se detallarán las características de cada dispositivo así como su función en el sistema de monitorización.

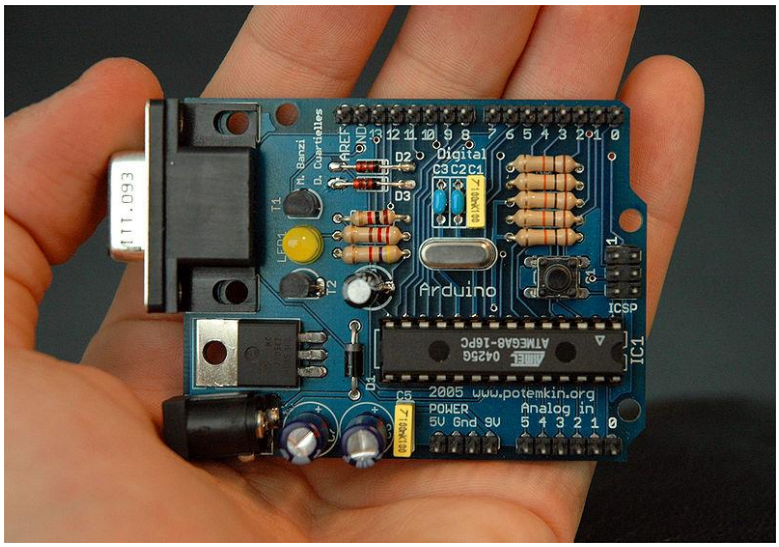


Figura 3. Placa Arduino RS232 (Wikipedia).

2.1.1. Nodo emonTx

La función del nodo ***emonTx*** es la de agrupar los datos recogidos por los distintos sensores. Está basado en un microprocesador **ATMEGA328P** usado ampliamente en proyectos ***Arduino***. En este caso, el diseño de la placa pertenece al proyecto **openenergymonitor** [3]. Se trata de un grupo de jóvenes cuya empresa está ubicada en Gales. Desde hace unos años se dedican a ofrecer soluciones para la transmisión de datos a través de **RF**. En este proyecto, está orientado a la recogida y transmisión de datos de sensores de temperatura y humedad. La **alimentación** de la placa se sitúa en **3.3V** por medio de **pilas/baterías** o mediante un cable **micro-USB**. Junto a la entrada USB se encuentra un regulador de tensión que se encarga de **rebajar** los **5V** introducidos por el puerto a los **3.3V** requeridos por el microcontrolador.

Esta placa cuenta con las siguientes conexiones:

- Conexión jack 3.5mm para sensores ***DS18B20*** [9].
- Conexión jack 3.5mm para sensor óptico (adaptado en este caso para admitir un sensor ***DHT11*** [10]).
- **Tres** conexiones jack 3.5mm para medir corriente **AC** mediante pinzas amperimétricas.
- Una entrada Jack especial para medir voltaje en una instalación de corriente alterna.
- Módulo **RFM12B** encargado de transmitir la información por radiofrecuencia.

La guía con los componentes, montaje y soldadura se encuentra en el siguiente enlace [4]:

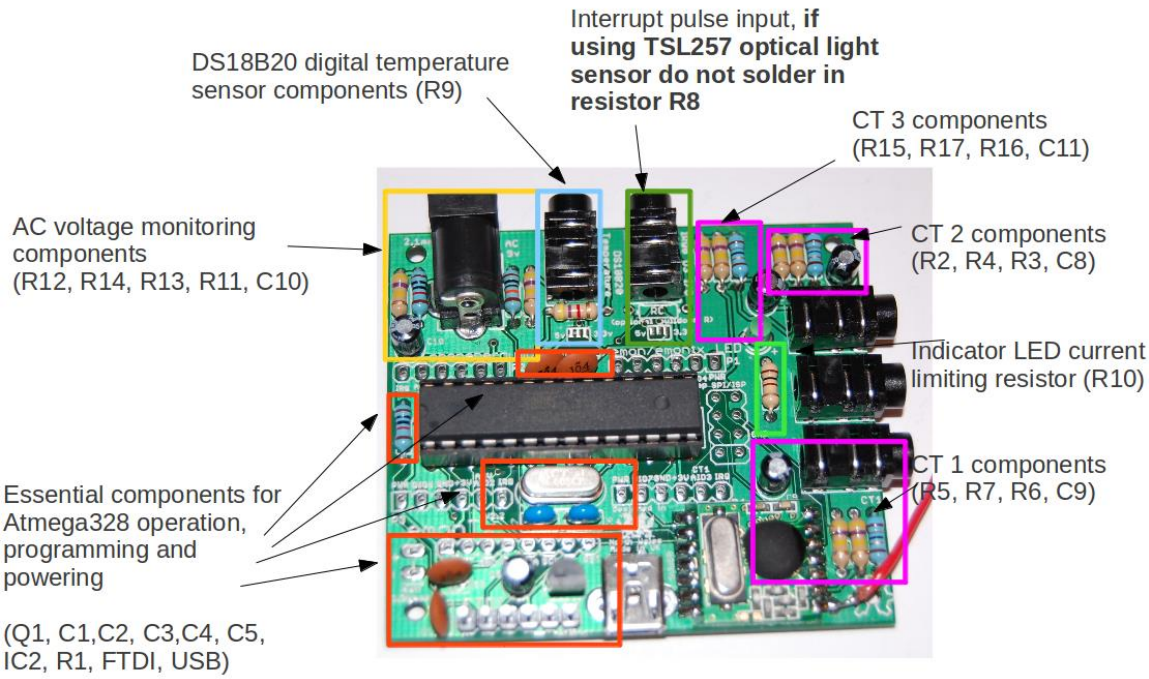


Figura 3. Nodo *emonTx* utilizado en el proyecto (openenergymonitor.org).

Para programar el microcontrolador **ATMEGA328** integrado en el nodo *emonTx* es necesario un cable ***FTDI***. Este cable establece una conexión entre el microcontrolador y el entorno de desarrollo (**IDE**) instalado en un ordenador. En este caso, el IDE "**Arduino IDE**" está instalado sobre el sistema operativo **Ubuntu**.

Figura 4. Cable FTDI utilizado para programar el microcontrolador.

2.2. Introducción a Raspberry Pi

El proyecto ***Raspberry Pi*** [2] comenzó como una plataforma dirigida a los colegios para que los más jóvenes aprendieran a programar pequeñas aplicaciones. Sin embargo, tuvo tanta acogida entre fanáticos de la electrónica que se dispararon las previsiones de venta. Su **bajo coste** y multitud de puertos entrada/salida lo hacen ideal para adentrarse en el mundo de la electrónica. El dispositivo se basa en un **SoC** (System-on-a-chip) Broadcom BCM2835 formado por un procesador **ARM11** a 700 MHz y **GPU** integrados.

Existen 2 modelos actualmente. Ambos modelos comparten mismo SoC y conexiones multimedia pero se diferencian en:

- **Modelo A**: 256 MB de RAM y un puerto USB.
- **Modelo B**: 512 MB de RAM, dos puertos USB y un puerto Ethernet 10/100.

Dispone también de varios puertos **GPIO, SPI, I²C, UART** para realizar pequeños proyectos. En la siguiente dirección se pueden encontrar algunos de estos ejemplos [11].

En este proyecto, la Raspberry Pi tiene un papel fundamental ya que es la encargada de recoger todos los datos que son enviados por los sensores, los almacena en una base de datos y los representa en la plataforma web. Debido al **bajo consumo** del dispositivo, es una buena opción para integrarlo en un sistema aislado donde el acceso a una fuente de energía puede ser difícil. Más adelante se detallará este consumo en varias pruebas realizadas pero, según la información oficial, el modelo B que es el usado en este proyecto tiene un consumo aproximado de **3.5W**; un consumo impensable hace años para un ordenador de propósito general como es este.

También mencionar que pueden instalarse varios sistemas operativos. El usado en esta ocasión es *Raspbian*, una distribución basada en *Debian* y ampliamente extendida.

A lo largo del documento se irán detallando las especificadores de esta SBC: Raspberry Pi.

Figura 5. Raspberry Pi (Wikipedia).

2.2.1. Módulo RFM12Pi

Este módulo es el encargado de recibir la información enviada por el nodo ***emonTx*** y, a su vez, transmitir dicha información a la ***Raspberry Pi*** [2]. La comunicación entre el módulo y la *RPi* se realiza mediante los puertos ***GPIO*** que dispone esta última. Un microcontrolador idéntico al del nodo decodifica la información y la convierte en una señal digital que es capaz de entender la ***Raspberry Pi*** [2]. Para la correcta lectura de los datos, se dispone de una serie de ***scripts*** o pequeños programas corriendo en segundo plano en el sistema operativo. Estos *scripts* leen y dan formato a los datos recibidos por los pines ***GPIO*** para su posterior almacenamiento en la **base de datos**. En cada operación, un pequeño ***LED*** parpadea para confirmar que recibe los datos correctamente.

El diseño de la PCB y sus características se pueden consultar en la página web del proyecto *openergymonitor* [5]. Básicamente se trata de un microcontrolador **ATMEGA328P** en formato SMD y un adaptador de RF, el anteriormente citado **RFM12B**. El módulo tiene el siguiente aspecto:

Figura 6. *RFM12Pi* vista superior.

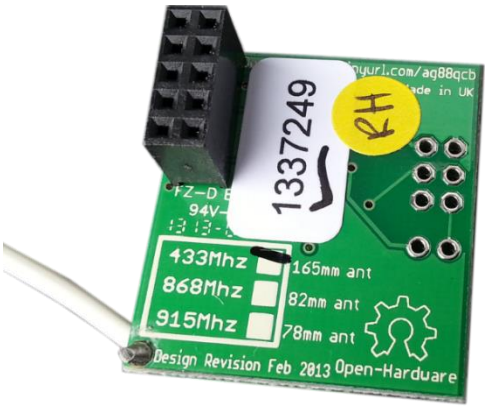


Figura 7. *RFM12Pi* vista inferior.

En la imagen superior se observa que el módulo puede trabajar en varias bandas de frecuencia, tan solo hace falta cambiar una juego de resistencias. La frecuencia usada en este proyecto es **868 MHz,** banda libre en Europa. Esta banda ofrece un rendimiento muy bueno para transmisiones de **medio alcance** y un consumo **reducido**. Además el uso de una antena más pequeña se agradece al contar con dispositivos de pequeño tamaño.

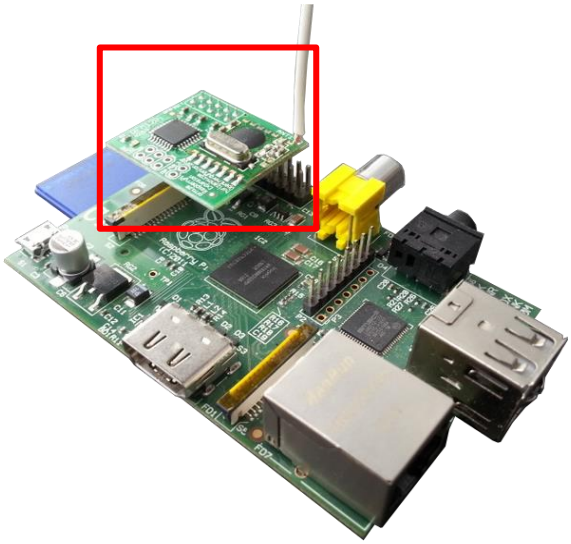


Figura 8. Instalación del módulo en la RPi.

El módulo, si se encarga en la tienda online, viene listo para usarse. La conexión con la RPi sería la siguiente:

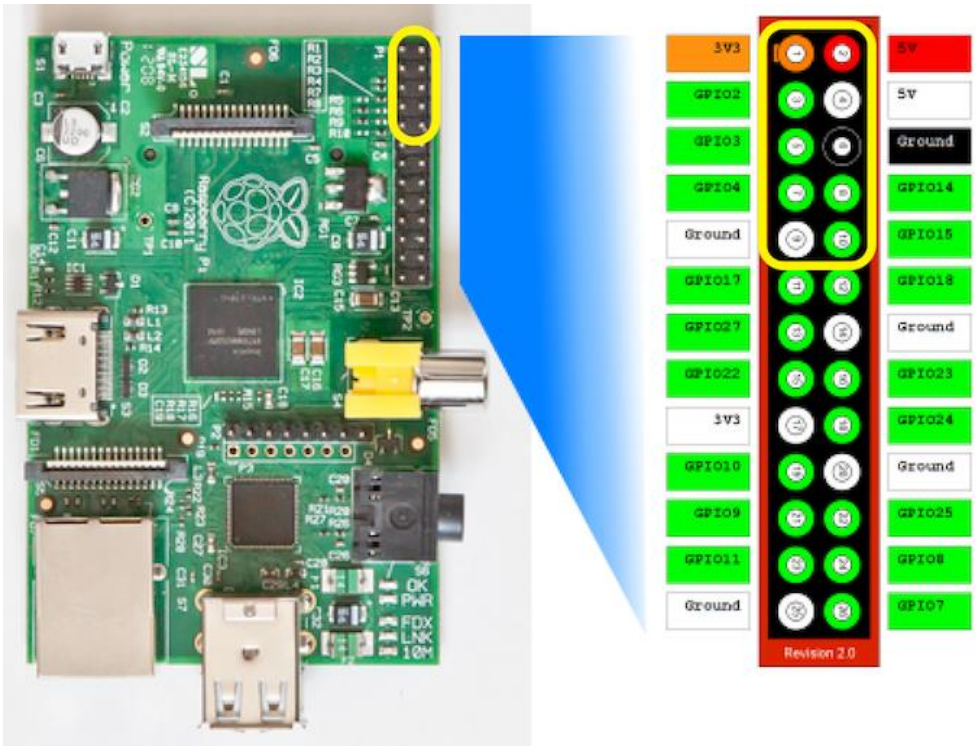


Figura 9. Conexión del módulo RFM12Pi en la Raspberry Pi.

Otra opción es adquirir los componentes necesarios de manera separada y encargar la PCB a un fabricante, ya que los diseños están publicados y se pueden replicar libremente.

El diseño de la PCB se rige por el siguiente esquemático:

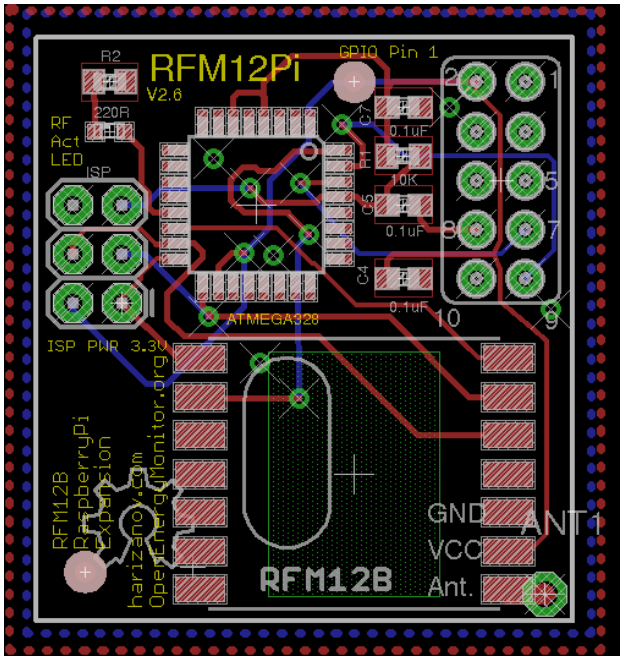


Figura 10. Esquemático del módulo RFM12Pi.

El módulo se basa en el transductor **RFM12B**, como se aprecia en la imagen superior. Este transductor, en formato SMD, es capaz de realizar comunicaciones en banda **ISM** con modulación **FSK**. Esto permite una rápida transmisión de datos con un consumo reducido. Además, la interconexión entre varios dispositivos iguales es inmediata si se cuenta con un controlador (en este caso el **ATMEGA328**) y permite trabajar a diferentes velocidades y frecuencias para adaptarse a los requerimientos del sistema.

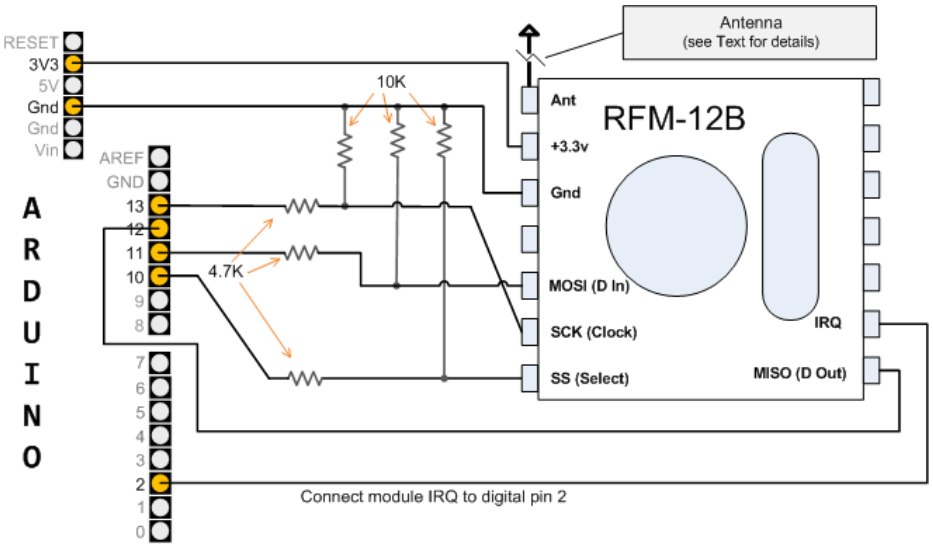


Figura 11. Conexión del módulo RFM12B a los pines de una placa Arduino

2.3. Sensores de temperatura y humedad

Para la medición de los factores ambientales se ha optado por un tipo de sensor barato pero lo suficientemente fiable y preciso para que los valores obtenidos sean adecuados.

2.3.1. DS18B20

El sensor utilizado para medir temperatura tiene como denominación ***DS18B20***. Se trata de un termómetro digital de alta precisión, entre 9 y 12 bits de temperatura en grados Celsius (el usuario puede escoger la precisión deseada).

Su temperatura operativa se encuentra entre **-50ºC** y **125ºC**, con una precisión, en el rango comprendido entre -10ºC y 85ºCde **±0.5ºC**.

Es un sensor económico pero bastante **preciso**. Su funcionamiento es **sencillo** y se puede implementar en multitud de proyectos. En este caso, se ha optado por escoger un modelo **aislado** y **sumergible** para que pueda soportar los agentes climáticos para una instalación al aire libre. Sin embargo, también existen las variantes en formato discreto para uso en placas de circuitos.

Estos sensores usan un protocolo de comunicación llamado ***1-Wire*** (One-Wire), diseñado por **Dallas Semiconductor**. Está basado en un bus, un maestro y varios esclavos en una sola línea de datos. A la hora de leer los datos se necesitan unas librerías accesibles en su página web.

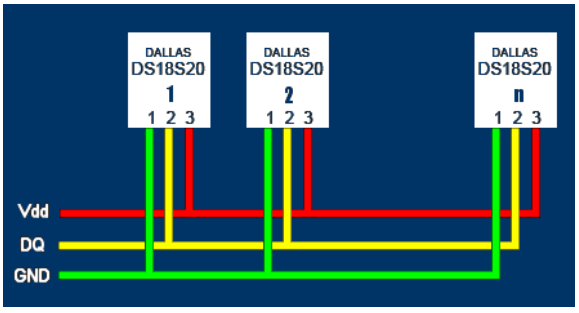


Figura 12. Pequeño esquema del protocolo 1-Wire.

Para la correcta captura de datos es necesario un ***burder resistor***, es decir, una resistencia entre la alimentación y el bus de datos para crear un ***pullup*** (señal digital de subida necesaria para la transmisión de datos) desde el pin de datos y VCC. En este caso, esta resistencia viene **incluida** en el nodo ***emonTx***.

Para más información, se puede consultar el datasheet referenciado más abajo.

Figura 13. Detalle del diagrama de conexión del sensor DS18B20.

2.3.2. DHT11

Este sensor se encarga de medir**temperatura** y **humedad**. Al igual que el sensor *DS18B20* tiene 3 líneas de conexión. El sensor ***DHT11*** individual viene soldado a una PCB específica que convierte los 4 pines a esos 3 pines junto a una resistencia ***pull-up*** y un condensador integrados. A continuación sus **especificaciones** y datasheet [0]:

- **Tensión** de funcionamiento: **3V – 5.5V**.
- Rango de valores del **20% al 90%** de **HR**.
- Rango de valores de **0ºC a 50ºC** de **temperatura**.
- **Resolución**: **1**, es decir, nos proporciona tan sólo **valores enteros** tanto de humedad relativa como de temperatura.
- **Precisión** de la **humedad** relativa: ±5%.
- **Precisión** de la **temperatura**: ±2ºC.

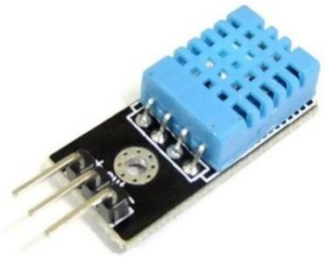


Figura 14. Detalle del sensor DHT11.

Como vemos, este sensor no ofrece grandes prestaciones y no tiene la precisión del DS18B20 pero sirve para obtener una **aproximación** coherente de los valores ambientales en el entorno donde se encuentra instalado el sistema. Este sensor también necesita unas **librerías** propias que se pueden encontrar fácilmente con una simple búsqueda. La tarea de estas librerías es crear una **interfaz** entre las **señales digitales** que manda el sensor al capturar los valores de temperatura y humedad y lo que recibe el microcontrolador para, posteriormente, mandar esos datos por RF.

La conexión es **sencilla**, pero para integrar el sensor en el nodo ***emonTx*** hace falta modificar una simple conexión en la placa como se detalla en el Anexo correspondiente.

Desde un punto de vista práctico, se ha decidido elegir este sensor para hacer una **comparación** entre los diferentes sensores y, con ello, observar las diferencias entre distintos métodos de obtención de datos.

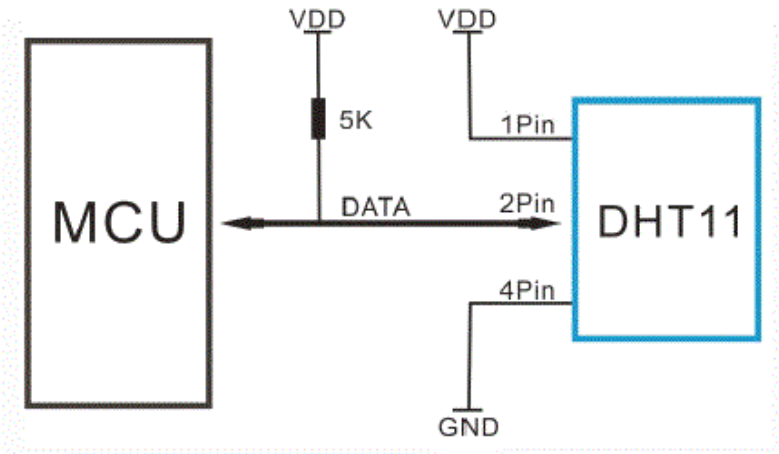


Figura 15. Conexiones del sensor *DHT11*.

2.4. Comunicación entre componentes

2.4.1. Radiofrecuencia

El principal problema que nos encontramos en un sistema con varios componentes es la comunicación entre ellos mismos. En este caso, el sistema se ha desarrollado para no depender de líneas físicas de alimentación. Para lograr este objetivo, el método más eficiente es el uso de radiofrecuencia. Supone la implementación de varios transductores dedicados a la comunicación pero se gana movilidad, seguridad y alcance. La frecuencia utilizada es 868 MHz, una banda de libre uso en Europa. La antena es de 82 mm para ¼ de longitud de onda.

2.5. Sistema de alimentación

Desde un principio, el sistema desarrollado en este documento se pensó para ser instalado en un lugar remoto, sin acceso a la red eléctrica tradicional. Por ello, un sistema basado en energías renovables es idóneo para alimentar el conjunto correctamente. En este caso, se ha optado por el diseño de una pequeña instalación fotovoltaica, suficiente para suplir la demanda energética exigida.

2.5.1. Placa solar

La placa elegida para este proyecto está fabricada con tecnología policristalina y con una potencia nominal de **5W**. Dispone de una salida de voltaje en bornes de **24V** y en la parte trasera se encuentra un regulador incorporado que proporciona **5V** mediante un puerto USB.

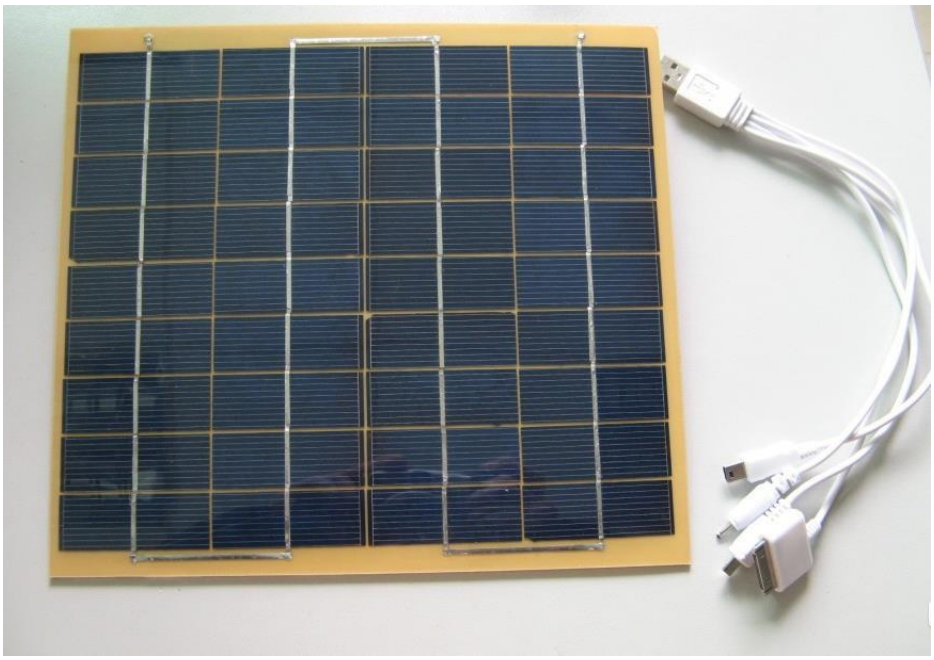


Figura 16. Placa solar vista frontal.

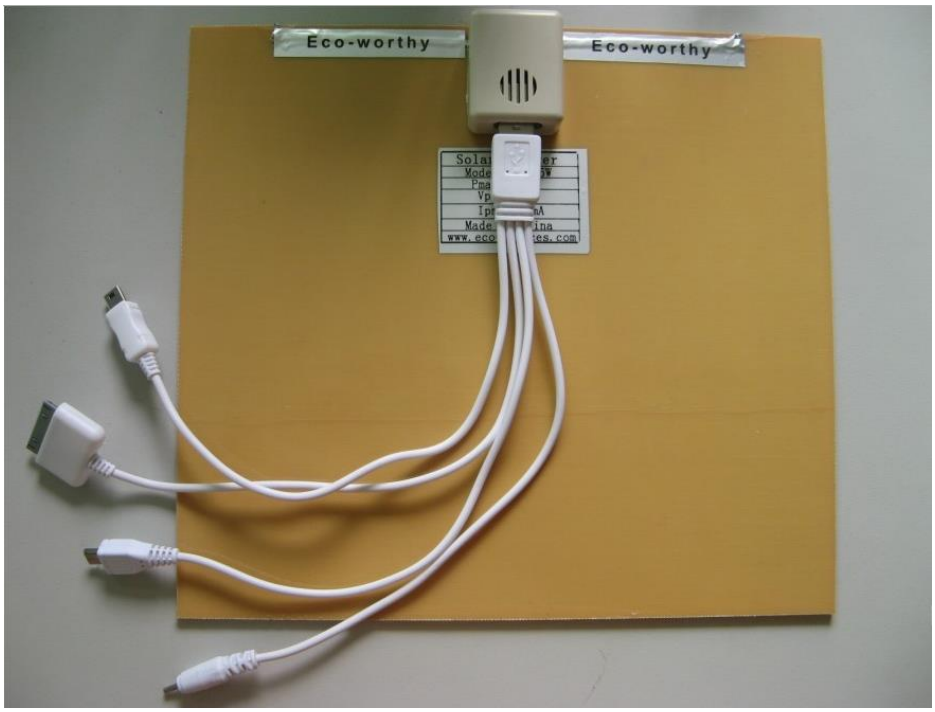


Figura 17. Placa solar vista trasera.

Una vez recibido el panel solar se realizaron varias pruebas para comprobar que, efectivamente, tanto el voltaje como la potencia suministrada era la correcta.

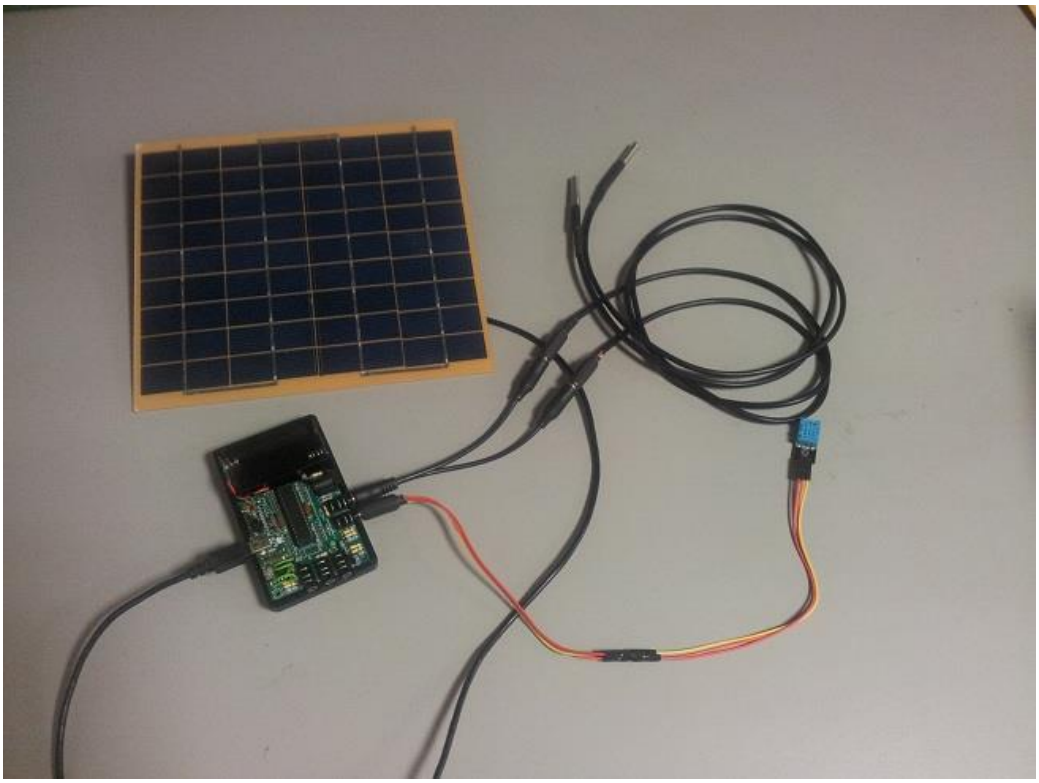


Figura 18. Primer montaje con el panel fotovoltaico.

2.5.2. Baterías (sistema de acumulación)

La elección del sistema de acumulación se ha basado en dos criterios: proporcionar la suficiente **autonomía** para que el sistema se mantenga operativo durante al menos un día sin sol incidente y, por otro lado, **reducir** al máximo la **circuitería** necesaria para la alimentación.

El tipo de batería elegido es el denominada **18650**, ampliamente utilizada en baterías de **portátiles** e incluso en **coches eléctricos**. El modelo utilizado en este proyecto se corresponde al de la imagen que se encuentra a continuación y su **tamaño** es bastante **superior** al de una pila alcalina AA. Estas baterías están fabricadas con la tecnología de Li-ion.

Su voltaje mínimo de operación es de **3.7V** y la máxima capacidad es de **3000mAh**.

Figura 19. Comparativa entre batería 18650 y pila alcalina AA.

Para este proyecto, se utilizará una batería de **4 pilas** configuradas en **paralelo**. Esta configuración ofrece una relación tamaño/autonomía bastante buena y es posible emplazarla en espacios reducidos.

Para regular la carga de las 4 baterías es recomendable contar con medidas de protección contra cargas/descargas indeseadas, como sucede a veces en baterías tipo Li-ion. Por ello, se ha decidido incorporar un regulador de carga específico para baterías 18650. Y, como medida de protección adicional se ha incluido un regulador de tensión al final de las baterías para que el nivel de voltaje sea siempre constante en la entrada de los dispositivos encargados de la monitorización.

A continuación se detallarán las especificaciones de ambos dispositivos.

2.5.3. Regulador de carga

Para el correcto funcionamiento de este tipo baterías es necesario contar con un regulador de carga. Este dispositivo se encargará de **mantener** la carga de las baterías a un **nivel constante** para que no haya desequilibrio entre ellas. Al tener **4 baterías en paralelo**, la suma de intensidades llega a la cifra de **12Ah** por lo que el uso de este regulador es **imprescindible**. Un ligero desequilibrio entre las cargas puede llegar a provocar **daños irreparables** en las mismas y en todo el sistema en general, ya que se puede dar el caso de que se incendien o incluso exploten.

En este caso, el regulador seleccionado se encuentra en una pequeña placa del tamaño de una tarjeta SD y está basado en un controlador **TP4056**. La información específica se puede encontrar en el datasheet correspondiente.

Figura 20. Detalle del controlador de carga.

En la siguiente gráfica podemos observar el tiempo que lleva cargar una batería de **1000mA** con una tensión VCC de 5V; y cómo va variando la relación **corriente/voltaje** durante el tiempo de carga. Para el caso que concierne en este proyecto sería multiplicar por **12** estos tiempos.

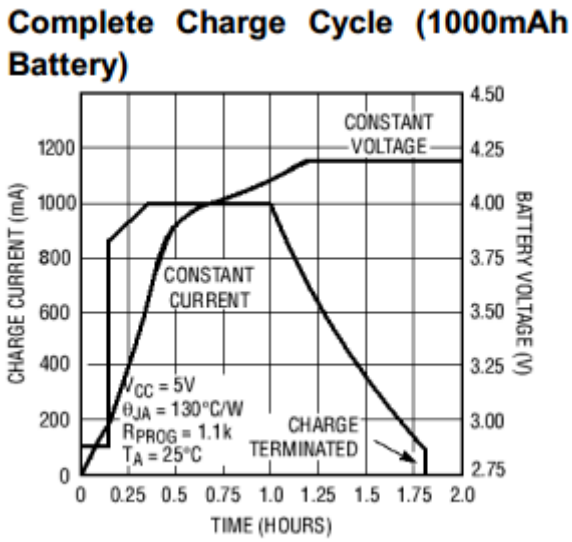


Figura 21. Gráfica del tiempo de carga.

2.5.4. Regulador de tensión

Debido a la tensión de salida de las baterías, no es posible alimentar adecuadamente el nodo *emonTx* ni la *Raspberry Pi* por lo que se necesita un regulador de tensión capaz de suministrar la cantidad de voltaje (constante) necesaria para alimentar estos dispositivos.

Como líneas de alimentación se puede usar la entrada USB o la conexión jack. El puerto USB soporta hasta 5V mientras que la entrada jack admite hasta 12V. Se puede apreciar que en la placa se encuentran dos reguladores de tensión: uno a 3.3V y otro a 5V. La elección de uno u otro se realiza con las pestaña amarillas de la imagen inferior.

Este doble regulador linealiza el voltaje que alimenta los componentes; es decir, evita un cambio brusco en el voltaje en caso de descarga de las baterías y eventualmente pueda llegar a dañar cualquiera de los componentes.

Figura 22. Detalle regulador de tensión doble

El regulador además cuenta con un interruptor para un apagado rápido, aunque no es recomendable abusar de él, ya que podría darse el caso de corromper el sistema de archivos de la RPi o incluso estropear algún elemento del sistema.

Ya que el regulador ofrece a su salida un valor constante a 3.3V, y con el objetivo de optimizar el rendimiento del sistema, cuando se conecta la alimentación a la RPi es posible realizar un bypass a la entrada micro-USB y pasar por alto el primer regulador de tensión que esta presenta. Este regulador integrado ofrece un pobre rendimiento y por lo tanto perjudica a la autonomía final de todo el sistema.

Dentro del conjunto de componentes que forman el sistema, la Raspberry Pi es el elemento que más consume, por lo que una pequeña mejora en su rendimiento repercutirá positivamente en el consumo total de energía.

3. Aplicaciones web y lenguajes de programación

3.1. Plataforma web (emoncms)

Se necesita una interfaz para representar los datos que se reciben de los sensores. En este caso se ha escogido un sistema de gestor de contenido (en inglés ***CMS***) ***emoncs*** [5], de licencia gratuita y libre distribución. Este *CMS* está diseñado específicamente para recoger los datos enviados desde la base de datos alojada en la *Raspberry Pi*.

Dentro de la plataforma existen varias secciones (explicadas en el Anexo correspondiente), con diferentes funciones dentro del sistema de monitorización. Una de las secciones más importantes es la de visualización de los datos.

3.1.1. Visualización de gráficas

El objetivo fundamental del presente proyecto es poder visualizar de una manera rápida e intuitiva los valores ambientales recogidos por los sensores. Una manera sencilla de hacerlo es superponiendo todos los datos recogidos y mostrándolos en una gráfica. De esta manera, podemos observar de un vistazo la temperatura, humedad y batería restante a lo largo del tiempo seleccionado. Podemos seleccionar los sensores a mostrar, el rango de tiempo, entre fechas señaladas, incluso la forma y escala de las propias gráficas.

A continuación se muestran ejemplos de gráficas integradas en la plataforma, así como conceptos de implementación de los módulos disponibles en la web.

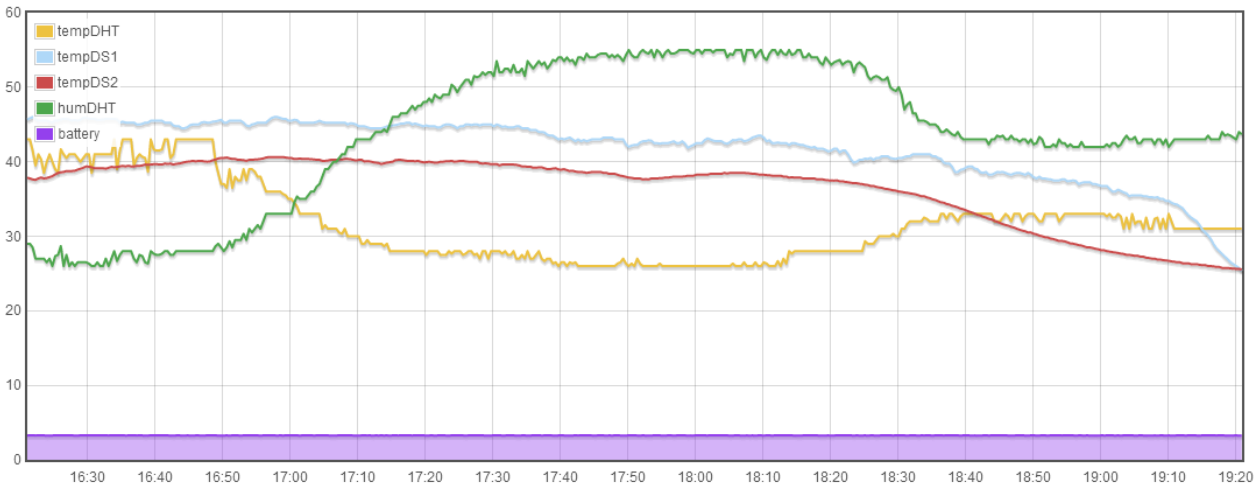


Figura 23. Gráfica mostrada en la plataforma web.

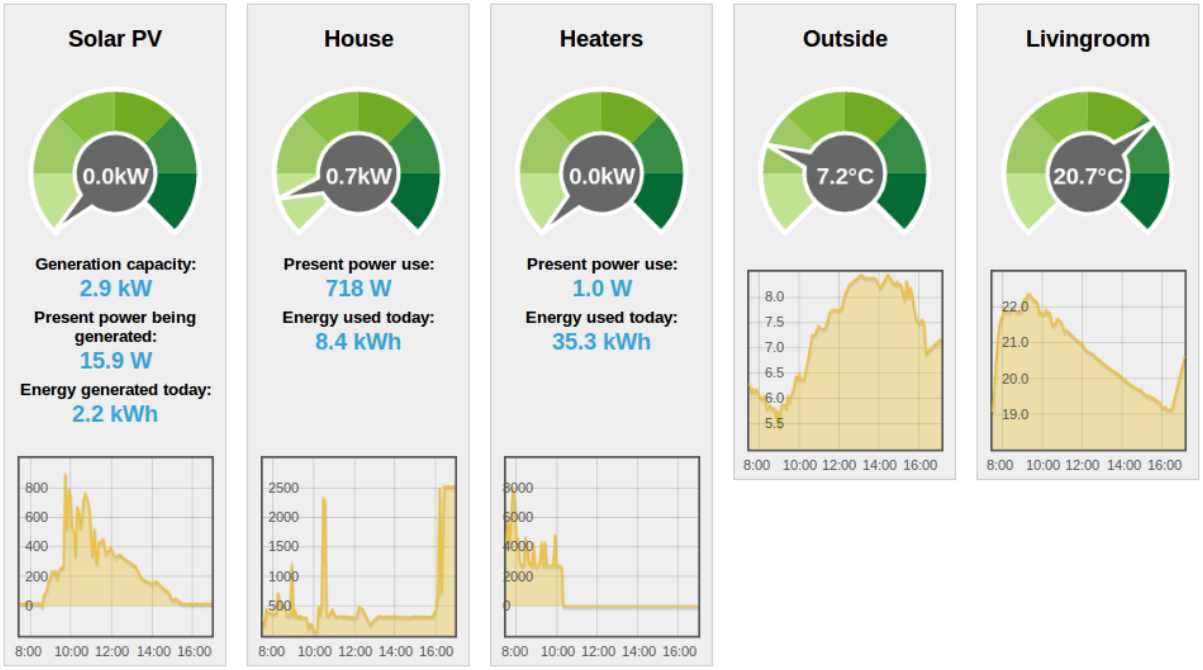


Figura 24. Concepto de implementación web en el hogar.

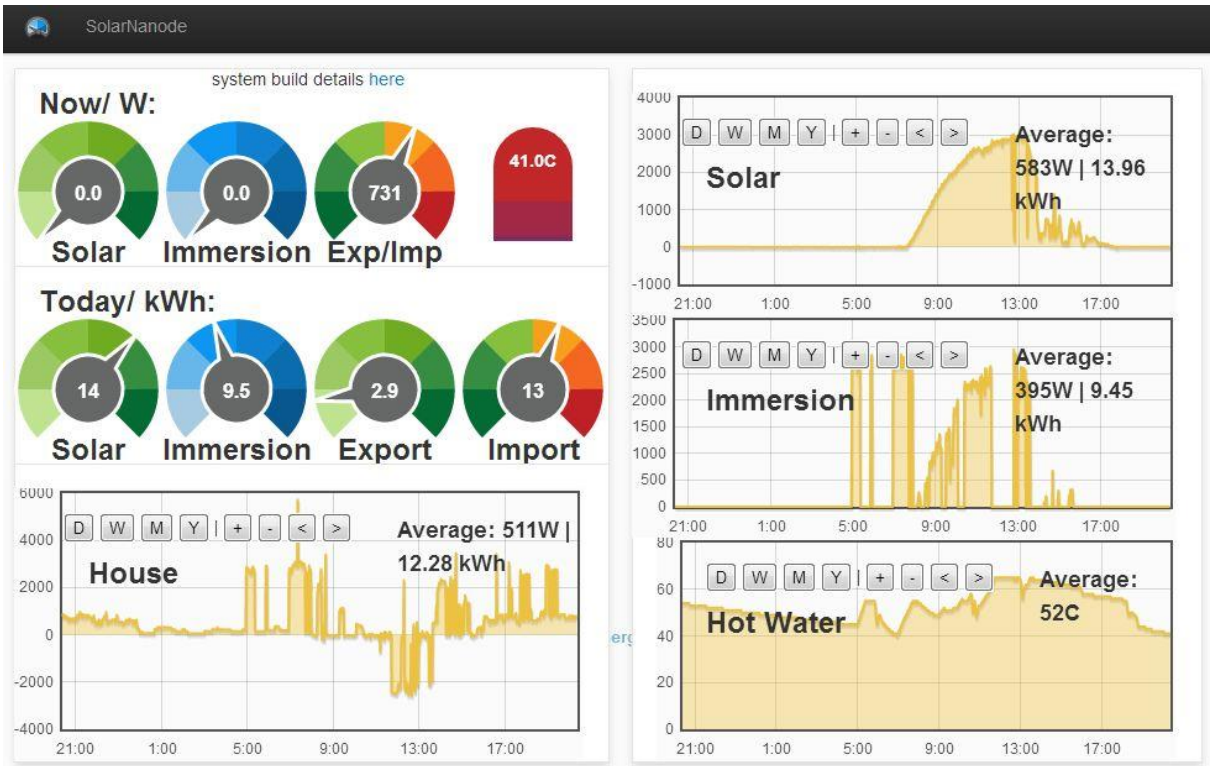


Figura 25. Concepto de implementación de un depósito de agua.

3.2. Lenguajes de programación

Durante la fase de preparación del proyecto se han usado una serie de "**scripts**" o conjunto de instrucciones necesarios para hacer funcionar correctamente todo el sistema. Estos "scripts" han sido desarrollados en un lenguaje extendido enormemente llamado **PHP** [14]. Este lenguaje es utilizado tanto en páginas "**web**" como en pequeños programas debido a su sintaxis **sencilla**, **limpia** y **fácil** de implementar y, también, al **rendimiento** que ofrece junto a otras plataformas como **bases de datos**, gestor de contenidos (**CMS**), etc. Esto es debido a que se trata de un lenguaje **interpretado**, es decir, cada instrucción se ejecuta **secuencialmente** cuando se inicia el "script"; al contrario de un lenguaje compilado donde un compilador es necesario para componer el archivo ejecutable que más tarde se iniciará.

Además, es completamente portable al no depender de la plataforma en la que se ejecuta, todo el sistema corre sin problemas entre plataformas, así que no dependemos de un único sistema operativo para controlar el sistema de monitorización.

Por otra parte, para alojar los datos recogidos por los sensores, se ha optado por una base de datos basado en "**MySQL**" [15], de código libre y gratuito. Alojando esta base de datos se encuentra un servidor "**Apache**" [16] corriendo en la **Raspberry Pi** [2]. Este servidor se encarga de dar soporte a las diferentes comunicaciones que son necesarias para tener acceso desde el exterior a los datos recogidos, así como, alojar la página web donde se representan dichos datos.

Figura 26. Lenguajes de programación utilizados.

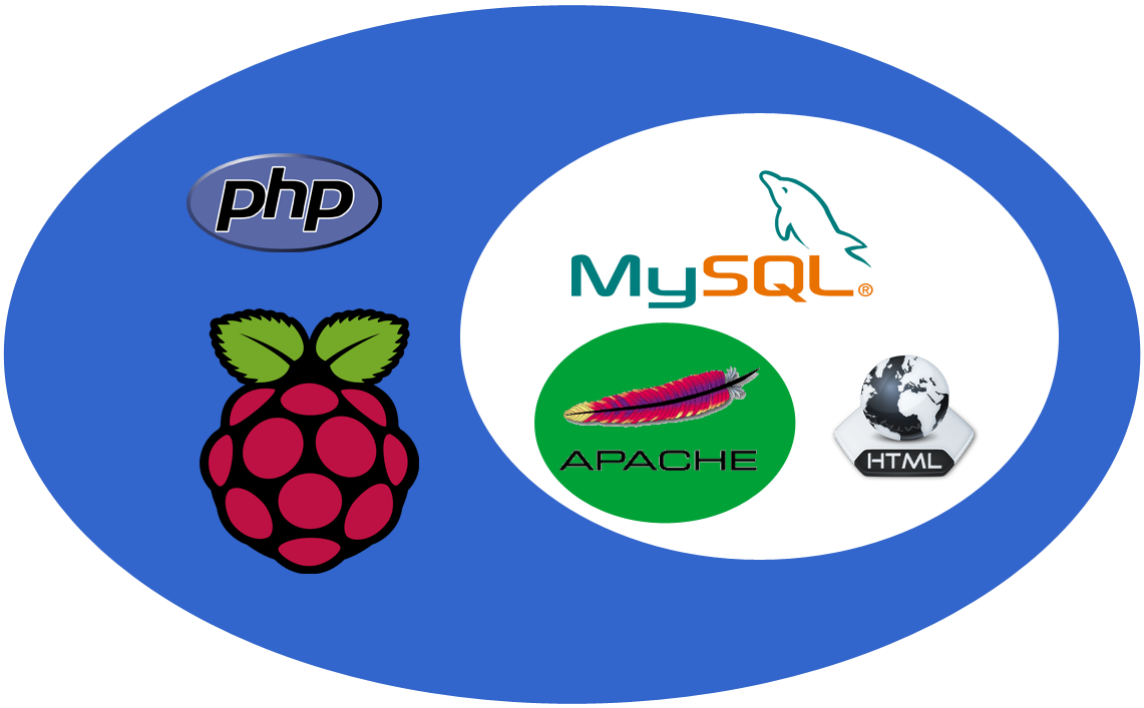


Figura 27. Configuración de lenguajes de programación.

Como se observa en la Figura 1, la configuración del almacenamiento de datos consta de varios "bloques". Desde el exterior al interior:

- Sistema operativo ***Raspbian***: basado en la distribución Linux *Debian* y optimizada para la Raspberry Pi.
- ***Scripts*** basados en *PHP*: necesarios para la comunicación entre el módulo RF y los GPIO, que convierten los datos recibidos por RF a información que procesa la RPi.
- Apoyando todo esto está el servidor ***Apache***, cuya función es soportar el servidor local donde se alojan los siguientes servicios:
    - ***MySQL***: base de datos relacional con todos los datos que se han ido recogiendo.
    - Plataforma *web* basada en ***HTML***: su función es la de crear una interfaz entre todos los elementos. Desde aquí se puede configurar la representación de los datos, tiempo de muestreo, etc.

Los otros servicios, módulos, etc. dependen de uno u otro bloque mencionados arriba. Por ejemplo, si el usuario decide cambiar el tiempo de muestreo, esa información pasa de un formulario de la plataforma web al servidor *Apache* que, a su vez, transmite esa información al *script* correspondiente que configura ese parámetro vía RF al nodo *emonTx*.

El sistema operativo es el encargado de tener todo bajo control y de establecer la prioridad para cada proceso. Por eso, un sistema liviano y con los servicios mínimos asegura que todo el sistema sea más fiable y su autonomía sea mayor.

4. Pruebas y resultados

Durante el transcurso del proyecto se llevaron a cabo una serie de pruebas experimentales para confirmar el correcto funcionamiento del sistema. Las pruebas se llevaron a cabo en el laboratorio del Grupo de Ingeniería Fotónica, dentro del Edificio de Telecomunicaciones de la Universidad de Cantabria.

4.1. Pruebas cámara climática:

La primera prueba que se realizó fue un programa de cambio de temperatura. Para ello, se introdujo la placa con los sensores incluidos en una cámara climática [5], la *Raspberry Pi* se situó en una sala adyacente, para comprobar la transmisión por radiofrecuencia y se procedió a iniciar el programa con los distintos rangos de temperatura. Dicho programa empezaba a **temperatura ambiente** (**20ºC**) e iba bajando de temperatura hasta alcanzar los **-3ºC** para luego subir gradualmente hasta 20ºC de nuevo. El siguiente rango de temperaturas baja hasta 0ºC y, posteriormente, sube en intervalos de **10ºC** hasta llegar a **50ºC**. La última bajada corresponde a la apertura de la puerta de la cámara para dejar reposar los sensores y observar su "recuperación".

Como más tarde se observará, se aprecia la **similitud** en los valores recogidos por ambos sensores. Sin embargo, se puede observar también las **limitaciones** del sensor *DHT11*, ya que no es capaz de bajar de **0ºC** en sus medidas.

En la siguiente imagen se muestra el nodo emonTx emplazado en la cámara climática.

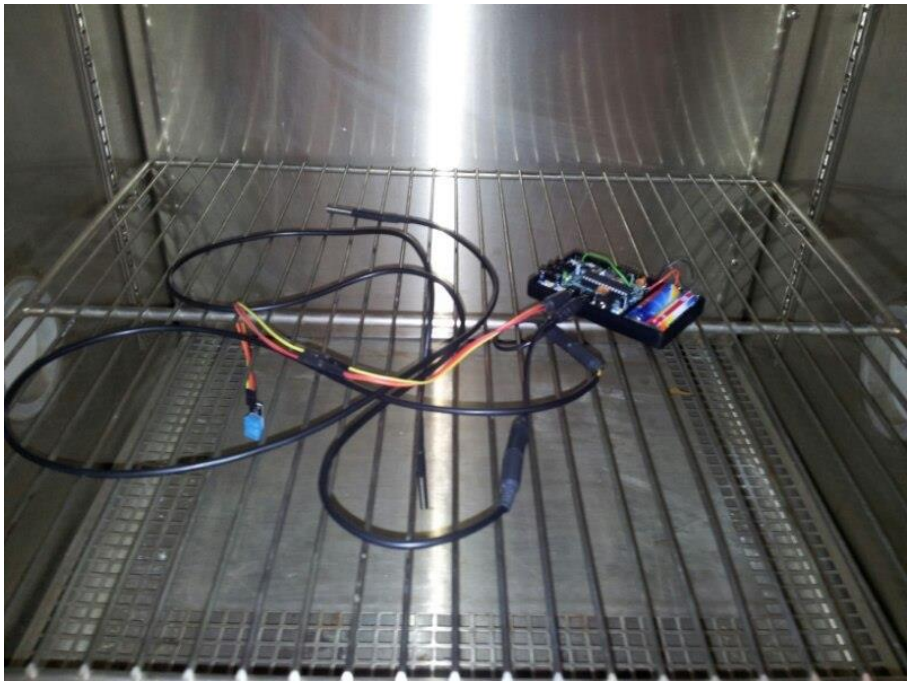


Figura 28. Placa con sensores dentro de la cámara climática.

A continuación se muestran algunas capturas de la prueba que se llevó a cabo.

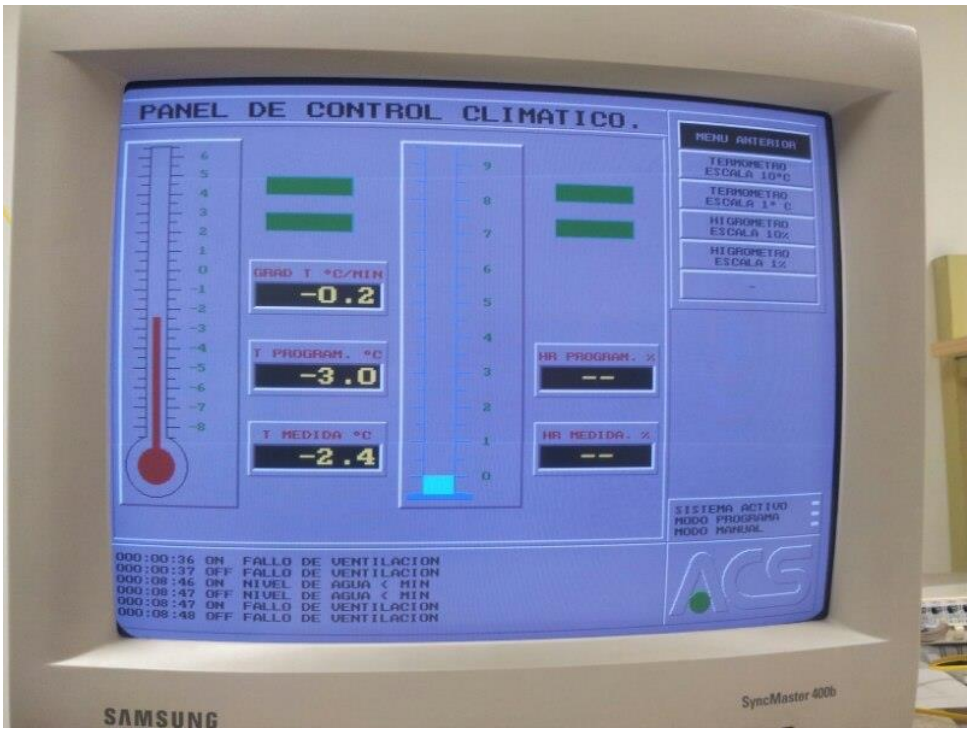


Figura 29. Detalle del funcionamiento de la cámara climática.

Figura 30. Segunda fase del programa en la cámara climática.

Las gráficas con los resultados se pueden ver a continuación:

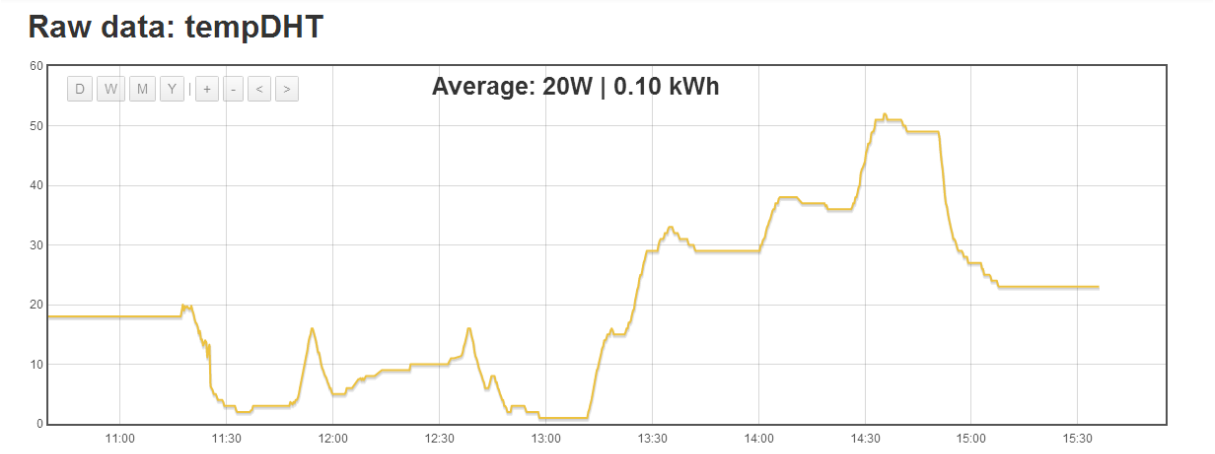


Figura 31. Detalle de la gráfica del sensor de humedad/temperatura (DHT11).

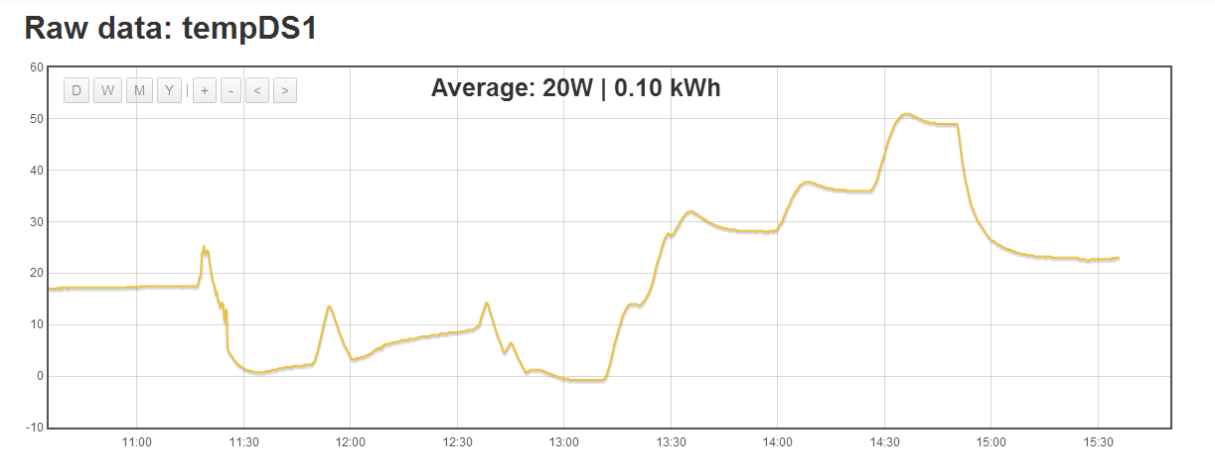


Figura 32. Gráfica del sensor 1 (DS18B20).

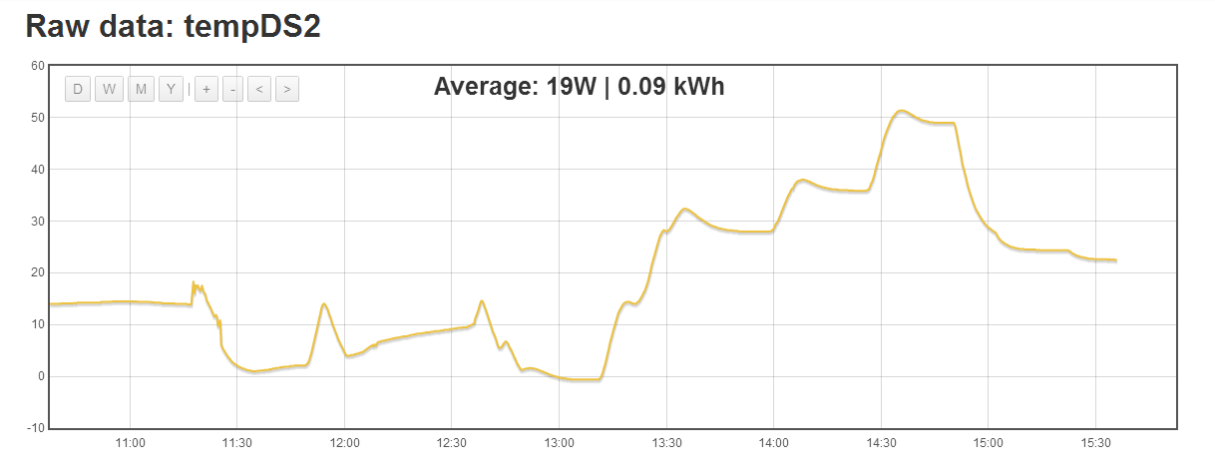


Figura 33. Gráfica del sensor 2 (DS18B20).

4.2. Mediciones en laboratorio

4.2.1. Mediciones con multímetro

En esta ocasión, se llevaron a cabo una serie de medidas con un multímetro para comprobar en primera persona el consumo del sistema de monitorización al completo para así dimensionar correctamente la alimentación mediante energía fotovoltaica. Para esta prueba se utilizó un multímetro “HP” y una fuente de alimentación. Mostradas en la siguiente imagen:

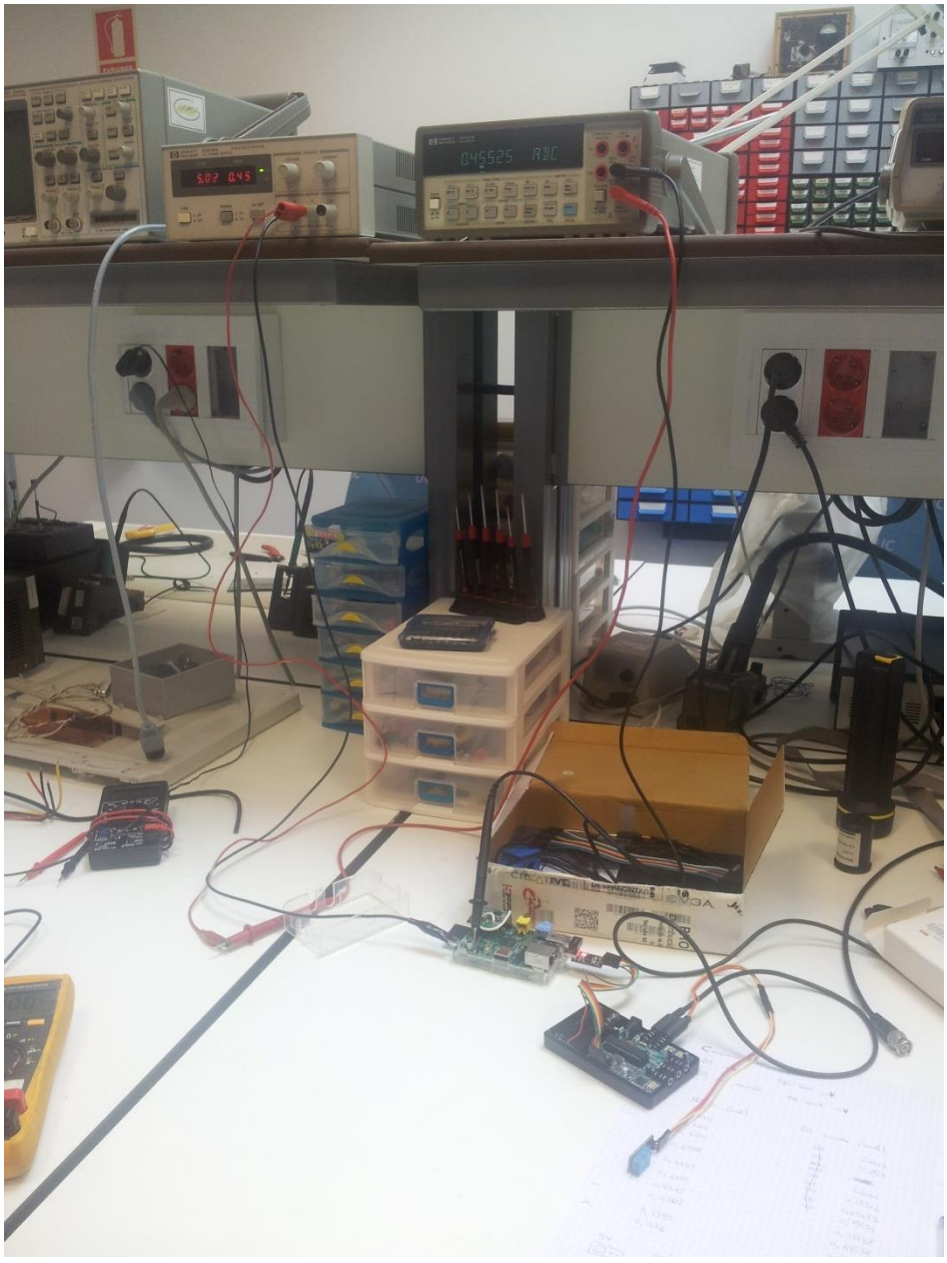


Figura 34. Mediciones en el laboratorio.

Como se observa en la imagen superior, se conectó la fuente de alimentación fijada a **5V** a la entrada de la ***RPi*** y, conectada a un puerto ***USB*** de esta, el nodo ***emonTx*** a través de un cable ***FTDI***. El multímetro se colocó después del regulador de tensión de entrada de la ***Raspberry Pi*** y se realizaron las medidas. Los resultados fueron los siguientes:

	Corriente (mA)	**Potencia (W)**
Medida 1	0.4310	2.2584
Medida 2	0.4314	2.2605
Medida 3	0.4305	2.2558
Medida 4	0.4308	2.2579
Medida 5	0.4335	2.2715
Medida 6	0.4302	2.2542
Medida 7	0.4312	2.2595
Medida 8	0.4305	2.2558
Medida 9	0.4395	2.3030
Medida 10	0.4316	2.2616
MEDIA	**0.4320**	**2.2638**

Tabla 1. Consumo de **RPi** alimentada a 5V.

	Corriente (mA)	**Potencia (W)**
Medida 1	0.4573	2.2865
Medida 2	0.4553	2.2765
Medida 3	0.4544	2.2720
Medida 4	0.4537	2.2685
Medida 5	0.4549	2.2745
Medida 6	0.4543	2.2715
Medida 7	0.4542	2.2710
Medida 8	0.4553	2.2765
Medida 9	0.4547	2.2735
Medida 10	0.4532	2.2660
MEDIA	**0.4547**	**2.2735**

Tabla 2. Consumo de **RPi**+**emonTx** sin transmit.

	Corriente (mA)	**Potencia (W)**
Medida 1	0.4756	2.378
Medida 2	0.4842	2.421
Medida 3	0.4763	2.381
Medida 4	0.5090	2.545
Medida 5	0.4857	2.428
Medida 6	0.5018	2.509
Medida 7	0.4705	2.352
Medida 8	0.4791	2.395
Medida 9	0.4771	2.385
Medida 10	0.4715	2.357
MEDIA	**0.4831**	**2.4154**

Tabla 3. Consumo de **RPi**+**emonTx** transmit.

	Corriente (mA)	**Potencia (W)**
Medida 1	21.405	70.636
Medida 2	21.399	70.616
Medida 3	21.408	70.646
Medida 4	21.403	70.629
Medida 5	21.418	70.679
MEDIA	**21.406**	**70.641**

Tabla 4. Consumo del nodo **emonTx** sin transmitir (sleep)

	Corriente (mA)	Potencia (W)
Medida 1	22.742	75.048
Medida 2	22.336	73.708
Medida 3	22.392	73.893
Medida 4	22.821	75.309
Medida 5	22.198	73.253
MEDIA	**22.497**	**74.242**

Tabla 5. Consumo del nodo **emonTx** transmitiendo

A continuación, los resultados de las pruebas con los sensores **DS18B20** y **DHT11**:

	Voltaje (V)
Medida 1	4.510
Medida 2	4.512
Medida 3	4.503
Medida 4	4.513
Medida 5	4.510
Medida 6	4.508
Medida 7	4.512
Medida 8	4.510
Medida 9	4.509
Medida 10	4.506
MEDIA	**4.509**

Tabla 6. Voltaje medido del sensor DHT11.

	Voltaje (V)
Medida 1	4.557
Medida 2	4.557
Medida 3	4.558
Medida 4	4.559
Medida 5	4.556
MEDIA	**4.557**

Tabla 7. Voltaje del DS18B20 en **HIGH**

	Voltaje (V)
Medida 1	4.514
Medida 2	4.510
Medida 3	4.518
Medida 4	4.514
Medida 5	4.521
MEDIA	**4.515**

Tabla 8. Voltaje del DS18B20 en **LOW**

Las mediciones de niveles ***HIGH*** y ***LOW*** se corresponden cuando el sensor envía los datos al microcontrolador y cuando está en reposo, respectivamente.

4.2.2. Mediciones con osciloscopio

En esta sesión de pruebas se llevaron a cabo algunas medidas para comprobar el comportamiento interno de los componentes que forman el sistema. En concreto, el reloj de operación del transductor **RFM12B**. Esta información nos ayuda a comprender cómo trabaja dicho transductor y cuál es el pico de consumo cuando realiza una transmisión de datos. En las siguiente imágenes se puede observar este funcionamiento; primero con la señal sin filtrar, es decir, donde se puede observar la implicación de los diferentes condensadores. Más tarde se midió la misma señal después de los filtros aplicados.

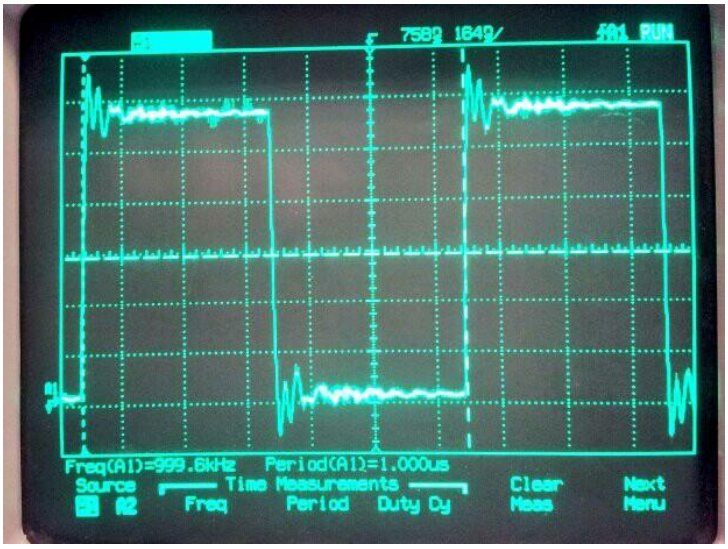


Figura 35. Medición del reloj de operación.

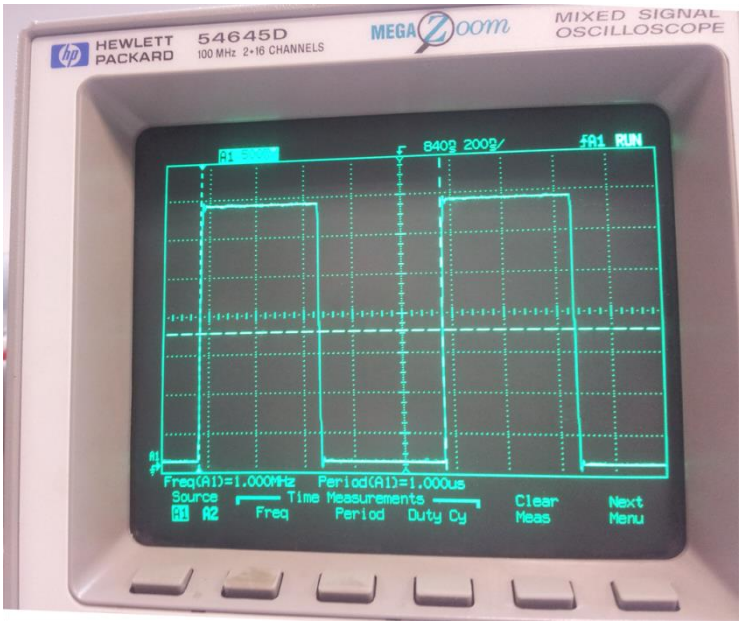


Figura 36. Medición de la señal reloj filtrada.

4.3. Funcionamiento al aire libre

4.3.1. Instalación y autonomía

Para una instalación al aire libre es recomendable contar con una carcasa o **protección** externa para los diferentes componentes. Los agentes climáticos pueden llegar a repercutir en el funcionamiento si no se toman las medidas pertinentes. Por ello, se ha optado por recubrir el nodo *emonTx* con una **carcasa** de plástico opaca, manteniendo intactas sus funcionalidades. Para el caso de la RPi, se cuenta también con una carcasa plástico pero esta vez transparente, distribuida por el fabricante.

El objetivo de estas medidas es la de proporcionar un lugar de trabajo adecuado, atendiendo a las condiciones climatológicas del entorno. Por ejemplo, según la zona en la que se trabaje se optará por un sistema u otro. No es lo mismo trabajar en zonas donde las precipitaciones son abundantes que en una zona árida o dentro de un sistema adyacente.

Figura 37. Carcasa del nodo emonTx.

Como se observa en la imagen superior, se ha dispuesto un "***holder***" para colocar 2 pilas AA.

Respecto a la autonomía, teniendo en cuenta los datos de consumo realizados anteriormente se puede llegar a la siguiente conclusión:

Si se optimizan las conexiones, los componentes utilizados y se fija el tiempo de muestreo en una medida cada 10 minutos, el sistema podría llegar a aguantar **un año y medio** con **2 pilas alcalinas AA**. Como el objetivo de este proyecto es proporcionar alimentación a todo el sistema, la alimentación necesita ser incrementada. Sin embargo, el sistema es capaz de funcionar sin luz solar durante al menos 12h seguidas gracias a la configuración de baterías utilizada.

5. Resumen, conclusiones y líneas futuras

5.1. Resumen

Hoy en día los sistemas de monitorización son fundamentales para controlar sistemas todo tipo: desde los más sencillos a los más complicados. En este proyecto se ha desarrollado un sistema de monitorización de bajo coste y fácil configuración para que cualquier usuario con curiosidad y mínimas nociones de electrónica pueda replicarlo para su propio caso.

Desde un principio, este sistema se ha diseñado para satisfacer un interés personal en el ámbito de la monitorización. Si bien en un primer momento se contó con otras alternativas, no fue hasta encontrar esta solución óptima que el autor desestimó las demás.

El uso de energías renovables es cada vez más y más necesario debido a la demanda energética del planeta. En este sistema de monitorización se ha hecho uso de este tipo de generación de energía para alimentar cada uno de los elementos que lo conforman. Con ello, se ha pretendido ofrecer independencia al conjunto de medios tradicionales de alimentación, con la posibilidad de emplazar el sistema en lugares remotos donde este tipo de alimentación es inviable.

Por otro lado, antes de empezar a desarrollar el proyecto, se llevó a cabo una intensiva búsqueda en el ámbito de sensores de bajo presupuesto. La elección de dichos sensores se produjo por varias razones: calidad/precio, soporte por parte de la plataforma y posibles implementaciones. En el caso que ocupa este documento, la sensórica utilizada ofrece un rendimiento inigualable. Y con la inclusión de dispositivos de bajo coste se abre la veda a nuevos proyectos derivados del principal.

Al ser un proyecto con motivación personal, el tiempo dedicado al mismo ha sido recompensado con su puesta en funcionamiento. La unión de varios componentes independientes entre sí formando un sistema en conjunto es emocionante. Cada elemento, cada sensor, está diseñado para funcionar en un entorno y región específicos, sin embargo, con la proliferación de las plataformas de hardware libre, es posible aunar todos estos elementos y formar un sistema.

Un sistema de monitorización económico, eficaz y eficiente, fiable y con el aliciente de usar energías renovables.

5.2. Conclusiones

Las conclusiones que se pueden sacar con este proyecto son varias. La primera, que es posible crear un sistema de monitorización simple, económico y fiable con el que controlar un determinado sistema. La segunda, que gracias a las plataformas de hardware libre existentes hoy en día y a la tremenda comunidad que da soporte a estas plataformas se proporciona un gran aliciente para adentrarse en el mundo de la electrónica.

En general, el proyecto desarrollado cumple las expectativas inicialmente propuestas. La implementación del sistema resulta sencilla e intuitiva. Para comprobar fehacientemente que el sistema funciona como debe, se realizaron varias pruebas de conexión, consumos, etc. para transmitir al lector que el sistema funciona y que funciona bien. El sistema podría ser usado para aplicaciones en: sistemas autónomos, ciudades inteligentes, etc.

5.3. Líneas futuras

Las posibles aplicaciones de este sistema no se quedan sólo en la monitorización de sistemas aislados fotovoltaicos. La configuración realizada en este proyecto sirve también para recoger datos en sistemas tan dispares como en depósitos de agua, gas, alimentos, etc. Al contar con varios sensores de temperatura y humedad relativa se podría controlar el proceso de conservación de cualquier tipo de sustancia; sin usar cables, a distancia y de forma segura.

Las aplicaciones que se presentan a continuación son diversas: cualquier sistema en el que se requiera cierto control necesita de una monitorización asociada. En este caso se ha orientado a magnitudes ambientales como son la temperatura y humedad, pero con los dispositivos adecuados, este sistema es capaz de diversificar sus funcionalidades.

Mirando al futuro, se pueden observar varias tendencias. El comercio colaborativo está en auge estos días; lo vemos cada día en noticias, periódicos, televisión, etc. Una manera de avanzar para una sociedad es crear y difundir conocimiento. Las comunidades que dan soporte a las plataformas usadas en este proyecto es lo que las mantienen vivas. Cada día, más y más personas sienten la curiosidad de crear algo por sí mismas. A lo largo y ancho del globo hay usuarios dispuestos a difundir su conocimiento para con los demás y así contribuir a mantener una comunidad sana y colaborativa; ya que si un usuario se siente respaldado por otros, éste a su vez prestará su ayuda a otro, y ese a otro…

Argumentando lo anterior, este proyecto sienta las bases para otro tipo de proyectos similares, que no iguales, puesto que cada usuario se encarga de adaptar su sistema a sus necesidades. Es por ello que en la comunidad académica este tipo de plataformas son muy útiles. Enseñar desde pequeños a fomentar la curiosidad, el conocimiento sobre temas que pueden sonar poco interesantes en un principio es importante. Ese era el principal objetivo cuando se formaron las plataformas *Arduino* y *Raspberry Pi*.

6. Glosario de términos

CMS: **Content Management System**, en español: 'sistema de gestión de contenidos'.

FTDI: Future Technology Devices International Ltd. es un fabricante de procesadores famoso por sus chips conversores USB-Serie. Las placas *Arduino* hacen uso de este dispositivo para conectar a diferentes tipos de ordenadores (fuente: http://arduino.cc/es/Main/Glossary).

FSK: **Frequency Shift Keying**, en español: modulación por desplazamiento de frecuencia

IDE: **Integrated Development Environment**, en español: 'entorno de desarrollo integrado'.

ISM: **Industrial, Scientific and Medical:** son bandas reservadas internacionalmente para uso no comercial de radiofrecuencia electromagnética en áreas industrial, científica y médica.

LED: **Light-Emitting Diode**, en español: 'diodo emisor de luz'.

RPi: Abreviación de Raspberry Pi.

SSH: **Secure SHell**, en español: 'intérprete de órdenes segura'.

USB: **Universal Serial Bus.** Protocolo diseñado para estandarizar la conexión de periféricos, como mouse, teclados, memorias USB, etc.

7. Bibliografía

[1] Plataforma Arduino: http://www.arduino.cc

[2] Fundación Raspberry Pi: http://www.raspberrypi.org/

[3] Openenergymonitor: http://openenergymonitor.org/

[4] Montaje: http://openenergymonitor.org/emon/emontx/make/assemble/buildguide22

[5] Emoncms: http://emoncms.org/

[6] EmonTx: http://openenergymonitor.org/emon/emontx

[7] EmonTx solderpad: http://solderpad.com/openenergymon/emontx/

[8] RFM12Pi: http://wiki.openenergymonitor.org/index.php?title=RFM12Pi_V2

[9] DS18B20: http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf

[10] DHT11: http://www.electrodragon.com/w/index.php?title=DHT11

[11] Proyectos Raspberry Pi (foro): http://www.raspberrypi.org/forums/viewforum.php?f=15

[12] Proyectos Arduino (inglés): http://playground.arduino.cc/Projects/Ideas

[13] Proyectos Arduino (español): http://playground.arduino.cc/Es/Projects

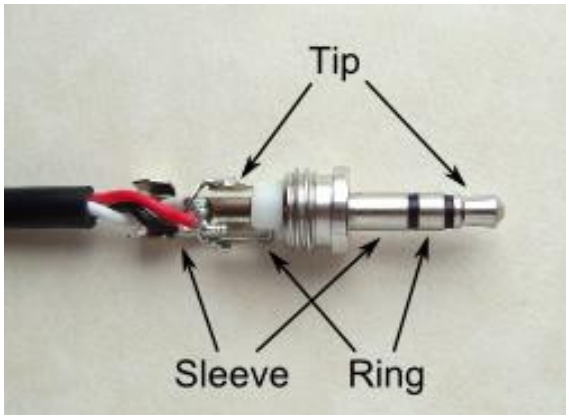
[14] PHP: http://www.php.net/

[15] MySQL: http://www.mysql.com/

[16] Apache: https://httpd.apache.org/

8. ANEXO I – Puesta a punto del sistema: sensores y adaptaciones

Para la unión de los sensores con los conectores jack que tiene el nodo *emonTx* es necesario soldar tres líneas. La de alimentación, tierra y datos. A continuación se muestra esta conexión.



Conexión de las líneas de conexión

Conexión	**Línea**
Tip	DQ (data line)
Ring	V_{dd}
Sleeve	GND

Como el nodo *emonTx* no está diseñado para soportar el sensor DHT11 hace falta hacer un pequeño cambio en la placa. Consiste en hacer un puente en una conexión, cerca del jack de entrada, para que haya una señal pullup de 5V.

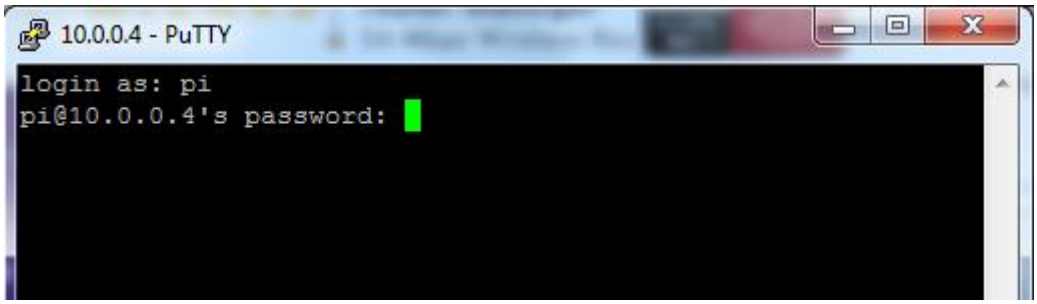
Detalle de la conexión puente

9. Anexo II – Instalación de RPi y de la plataforma web

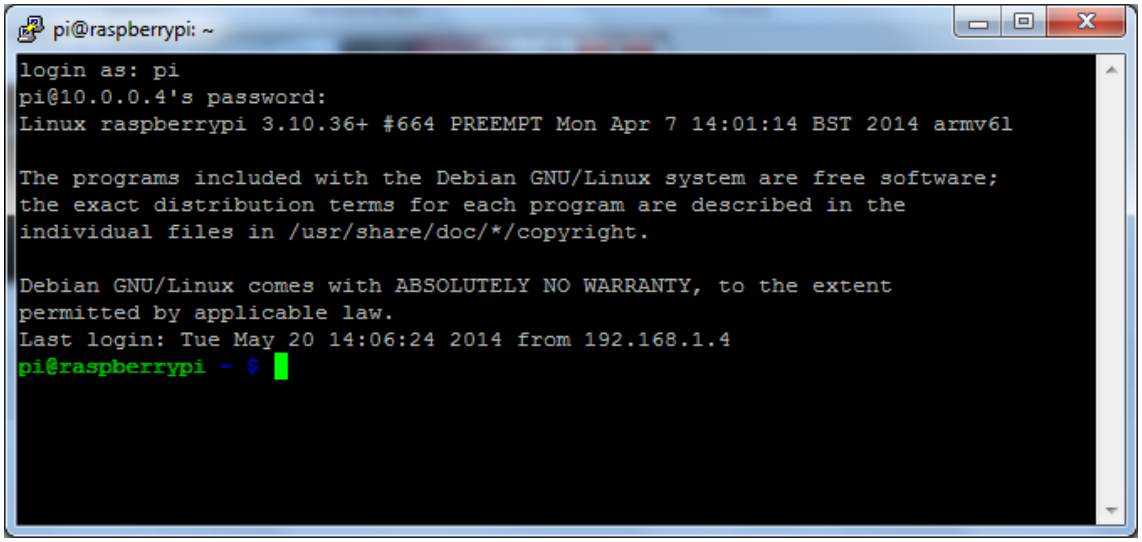
En primer lugar descargamos la última versión de Raspbian de la página oficial de la Fundación Raspberry Pi (http://www.raspberrypi.org/downloads/). A continuación introducimos una SD en el lector correspondiente y localizamos su ruta. En este caso es "/dev/sdb". El comando para grabar la imagen de Raspbian a la SD es el siguiente:

```
pc@pc-laptop:~/Descargas$ sudo dd bs=4M if=2014-01-07-wheezy-raspbian.img of=/de
v/sdb
[sudo] password for pc:
```

Donde "bs=" es el tamaño del bloque de escritura, "if=" el archivo a grabar y "of=" la ruta de destino. Una vez instalado el sistema operativo, el siguiente paso es configurar la RPi. Para ello, se puede conectar a un monitor o TV a través del puerto HDMI o, si está disponible una conexión Ethernet y sabemos la IP, podemos conectarnos a través de SSH. El usuario por defecto es "pi" y la contraseña "raspberry". Más tarde se podrá cambiar estos datos.

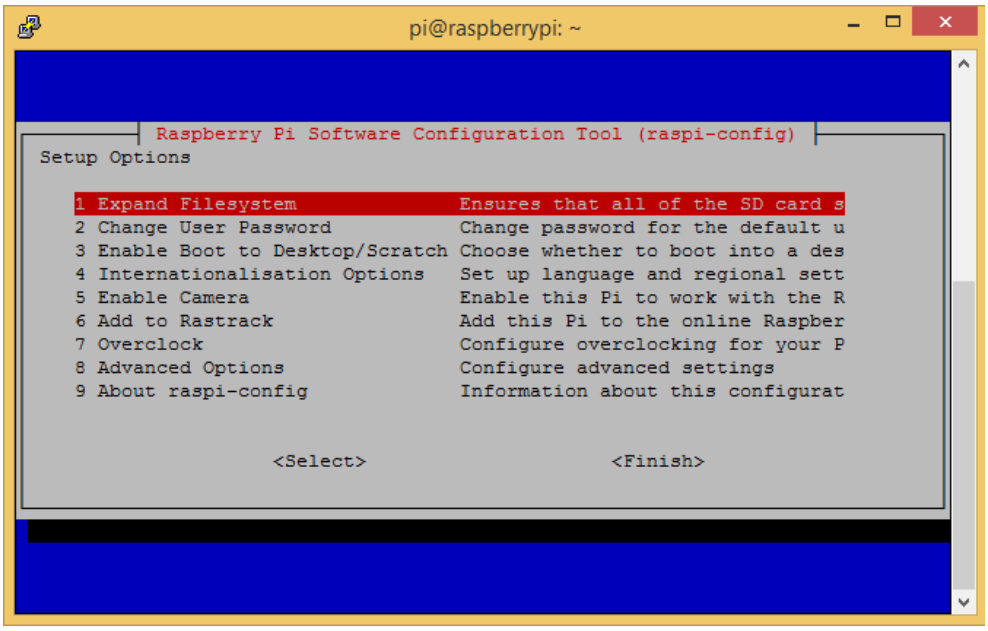


Una vez introducidos los datos de acceso debería salir una ventana como esta:



En la primera configuración es necesario ejecutar el siguiente comando para finalizar la instalación. Hay que seleccionar la opción 1.

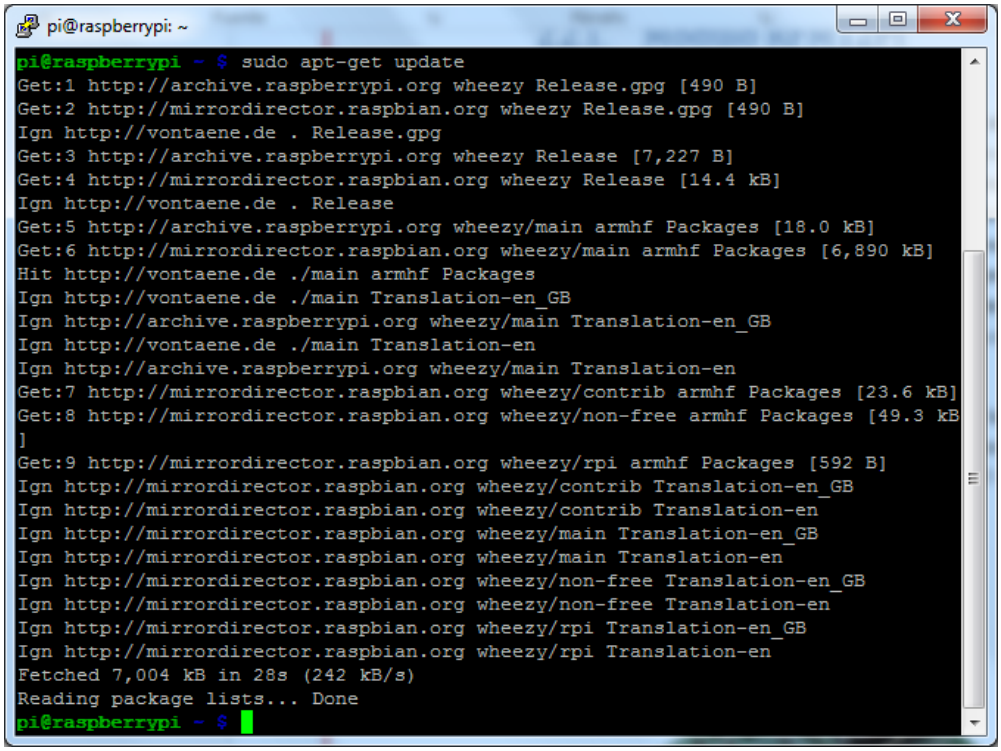
```
sudo raspi-config
```



También se puede configurar la contraseña de usuario, el modo de arrancar en el inicio (modo consola, modo escritorio) Cambiar el idioma, la disposición del teclado, etc. Para tener un sistema actualizado y evitar posibles errores con versiones anteriores de programas es útil actualizar los repositorios del sistema. Una forma rápida y sencilla es introducir la siguiente línea de comando:

```
sudo apt-get update && apt-get upgrade
```

Estos dos comandos actualizarán todos los programas instalados a sus últimas versiones estables.



El siguiente paso es instalar los programas y servicios necesarios para que la plataforma funcione como debiera.

```
sudo apt-get install apache2 php5 libapache2-mod-php5 php-gettext mysql-server mysql-client php5-mysql
```

Estos comandos instalan el servidor *apache*, el servidor donde se aloja la base de datos y las librerías *php* necesarias para la interconexión con las base de datos y la plataforma web. En este paso nos pedirá que ingresemos una contraseña para el usuario "*root*" de la base de datos. Es necesario recordarla ya que será necesaria más adelante.

Reiniciamos el servidor apache:

```
sudo /etc/init.d/apache2 restart
```

A continuación creamos la base de datos. Para ello, introducimos los siguientes comandos:

```
sudo mysql -u root -p
```

Introducimos la contraseña establecida anteriormente y continuamos.

```
mysql> CREATE USER 'emoncms'@'localhost' IDENTIFY BY 'password'
mysql> CREATE DATABASE emoncmsdb;
mysql> GRANT ALL PRIVILEGES ON emoncmsdb.* TO emoncms@localhost
mysql> \q
```

Lo primero que hace esta serie de comandos es crear un usuario en la base de datos y después crear una base de datos asignada a dicho usuario con todos los privilegios.

El próximo paso es descargar el sistema de gestor de contenido (CMS). Primero se debe instalar el cliente *git*, necesario para descargar los archivos fuente.

```
sudo apt-get install git-core
cd /var/
sudo chown $USER www
cd www
sudo git clone https://github.com/emoncms/emoncms.git
git pull
```

Nos aseguramos que es la última versión.

```
cd /var/www/emoncms/
cp default.settings.php settings.php
nano settings.php
```

Ahora es necesario introducir los datos de la base de datos:

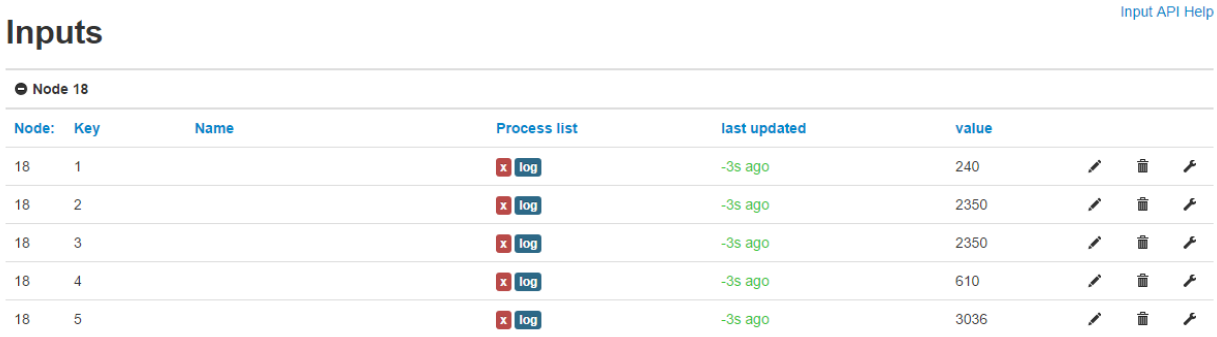
```
$username = “emoncms”;
$password = “password”;
$server = “localhost”;
$database = “emoncmsdb”;
```

Y presionamos Ctrl+X, Y para salir.

Con estos pasos, estaría montado todo el entorno para que la plataforma web funcione correctamente.

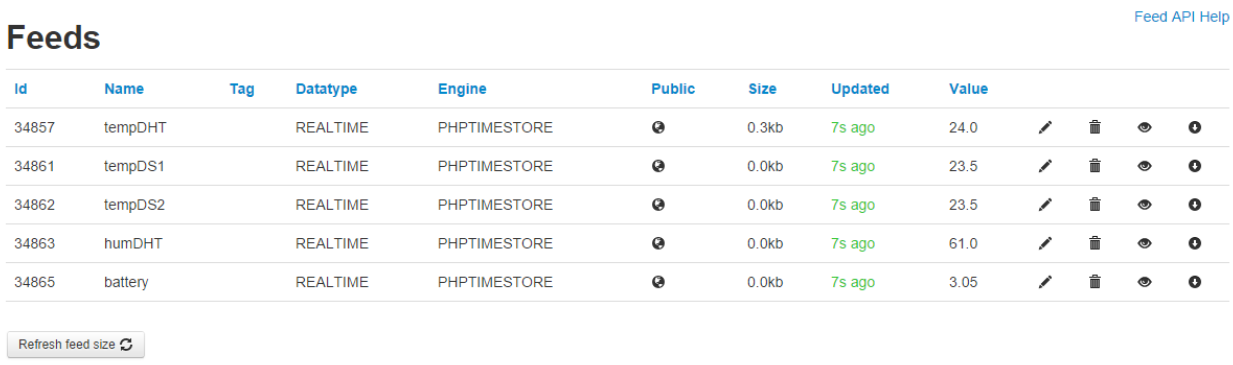
10. Anexo III – Añadir nodos, representar gráficas, etc.

Dentro de la plataforma web utilizada se puede observar en tiempo real el estado de los sensores: si están activos, su nombre identificativo y sus valores actuales. A través de las diferentes pestañas que aparecen se puede acceder a las diferentes funcionalidades del sistema. Por ejemplo, se puede modificar el formato de representación de los datos obtenidos, se pueden realizar operaciones como sumas, restas, medias, etc. para un análisis más concienzudo. En este caso, se mantendrán los datos lo más independientes posible.



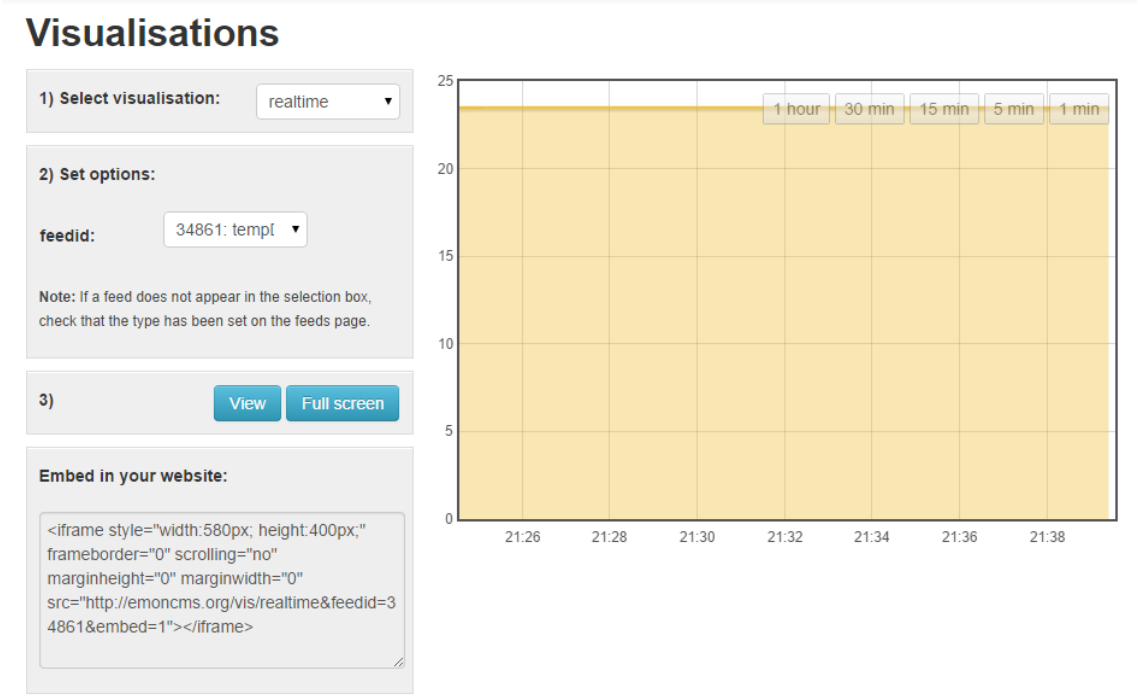
Sección de inputs (entrada de datos)

En la pestaña de “feeds”, podemos encontrar la configuración final antes de ser mostrada en las gráficas. En esta sección ya aparecen los datos en el formato deseado junto al tipo de valor, almacenamiento en base de datos y tamaño en la misma.

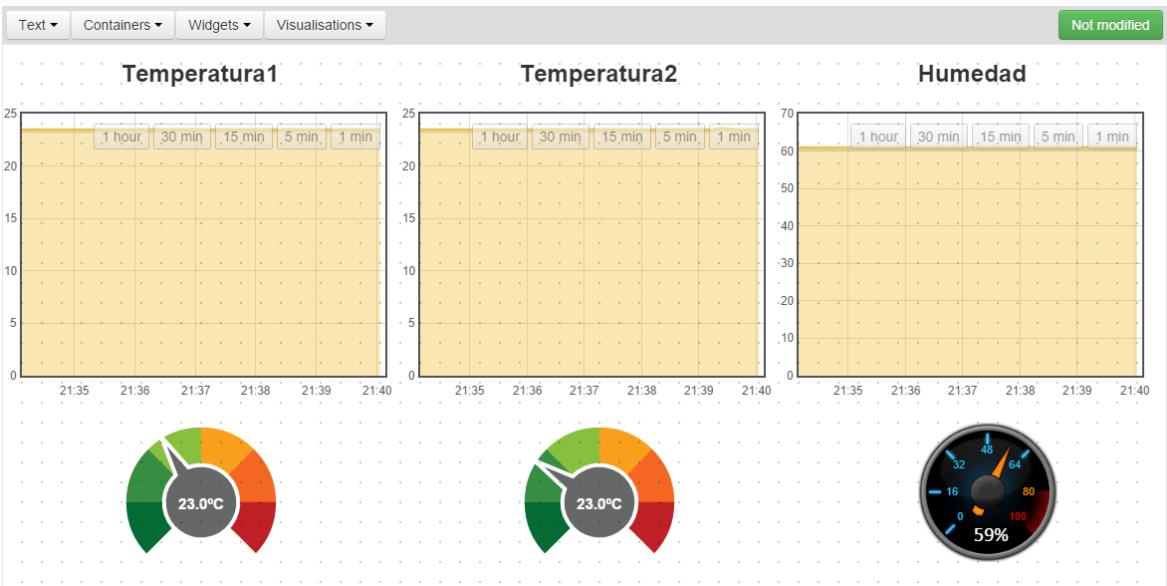


Sección de feeds (datos con formato)

La siguiente pestaña interesante es la de “visualisations”. En esta sección se puede configurar el aspecto que van a tener las gráficas y, también, observar de un vistazo los datos recogidos.



Por último, tenemos la pestaña de “dashboard” donde se permite elegir el tipo de gráfico a emplazar, modificar el tamaño y otras personalizaciones que permitirán crear un ecosistema intuitivo y atractivo al usuario.



Dashboard

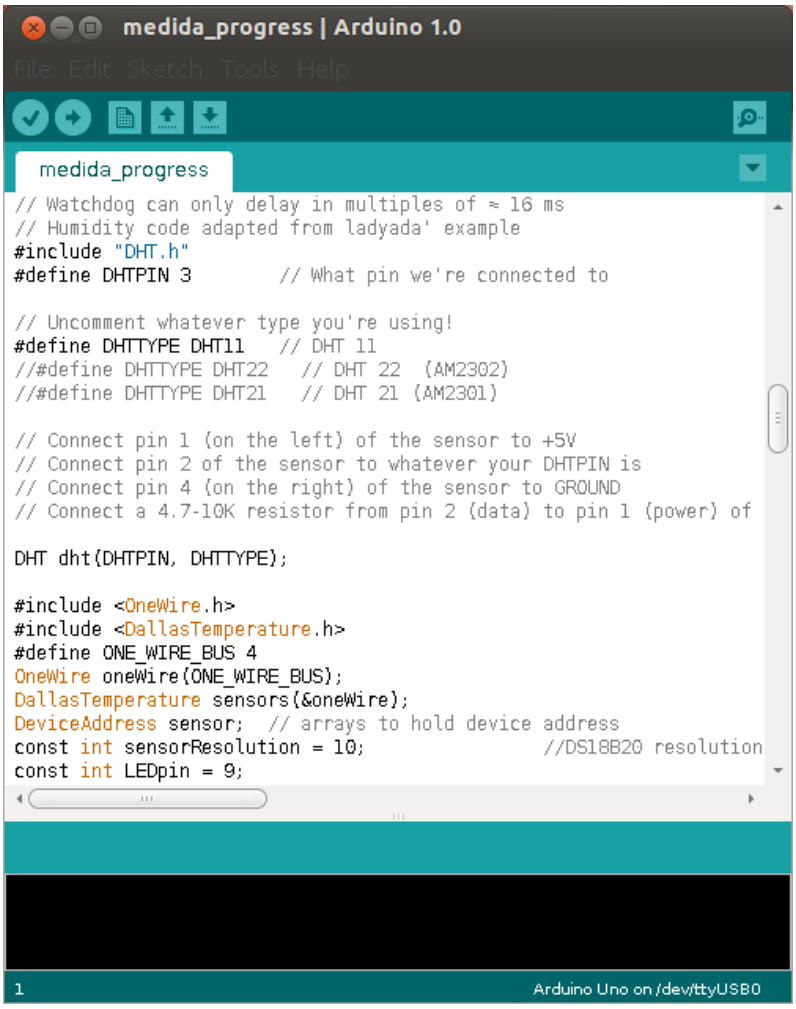
11. Anexo IV – Arduino IDE y otros scripts

11.1. Arduino IDE

Para programar el microcontrolador y que realice sus operaciones es necesario utilizar un entorno de desarrollo que permita cargar una serie de funciones en la memoria interna del chip. En este caso, se ha utilizado “Arduino IDE”, un entorno específico para placas Arduino y que cuenta con todas las herramientas necesarias para programar el microcontrolador.

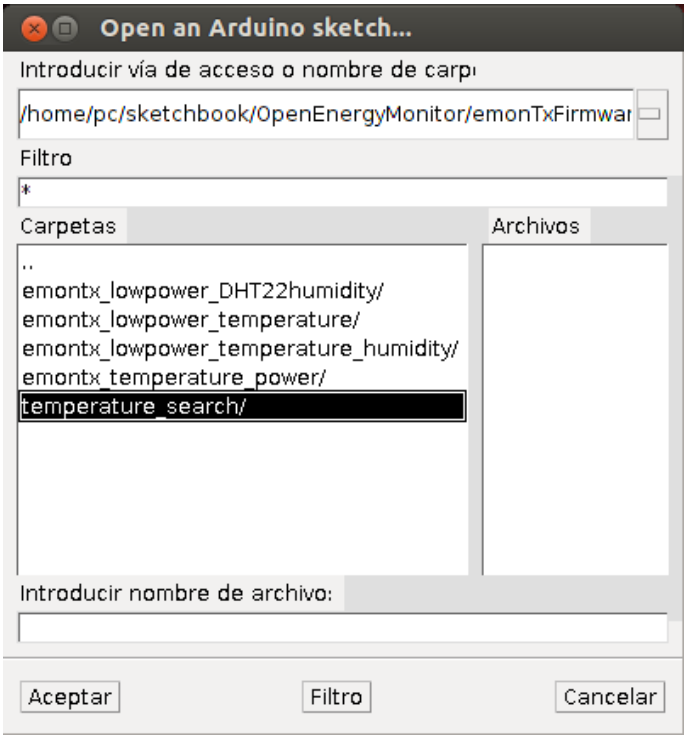
El uso de librerías se hace necesario ya que cada fabricante tiene una escala o rango de valores diferentes para una misma magnitud. Por tanto, para el sensor DS18B20 se usa las librerías “OneWire.h” y “DallasTemperature.h”, mientras que para el DHT11 se usa la librería “DHT.h”

En la siguiente imagen se puede observar la implementación de estas librerías.

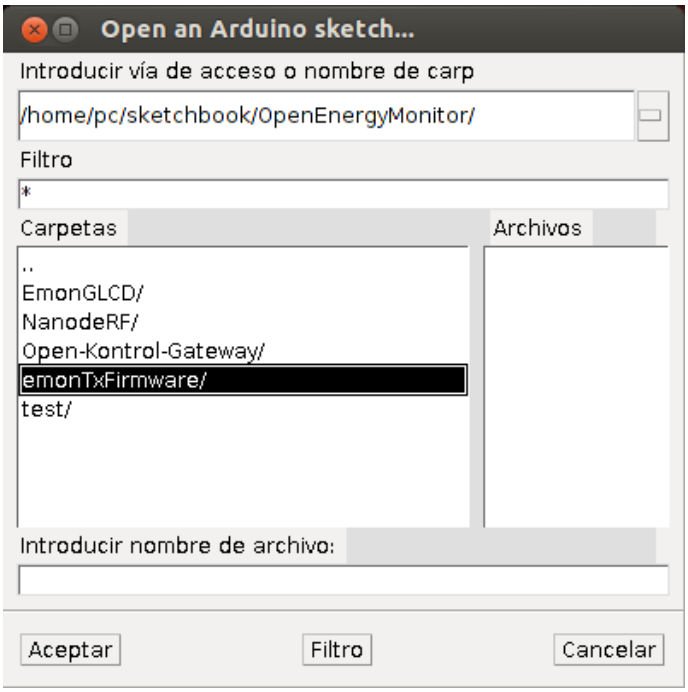


Entorno de desarrollo (IDE) “Arduino IDE”

Desde la página del proyecto "*openenergymonitor*" se ofrecen una serie de ejemplos para configurar los diferentes parámetros que ofrece el nodo *emonTx*. Uno de estos ejemplos consiste en recoger la dirección física de cada sensor DS18B20 conectado. Esto tiene utilidad cuando se conectan varios sensores y se quieren caracterizar. Gracias al bus One-Wire es posible conectar hasta 16 sensores bajo una misma línea de datos.



Ejemplo de búsqueda de dirección física de sensores DS18B20.



El usuario puede aprovechar estos ejemplos para adaptar su sistema de monitorización a sus propias necesidades: el tiempo de muestreo, la ID del nodo, etc.

11.2. Script Twitter

Un componente esencial de este proyecto es el social. Por ello, está incluido en el sistema un pequeño script que consiste en el envío de un *tweet* cada hora (si se dispone de conexión a Internet) con la temperatura, humedad relativa y el tiempo de captura. Lo que se presente con este servicio es dejar constancia e incitar a otros usuarios a consultar y usar el sistema.

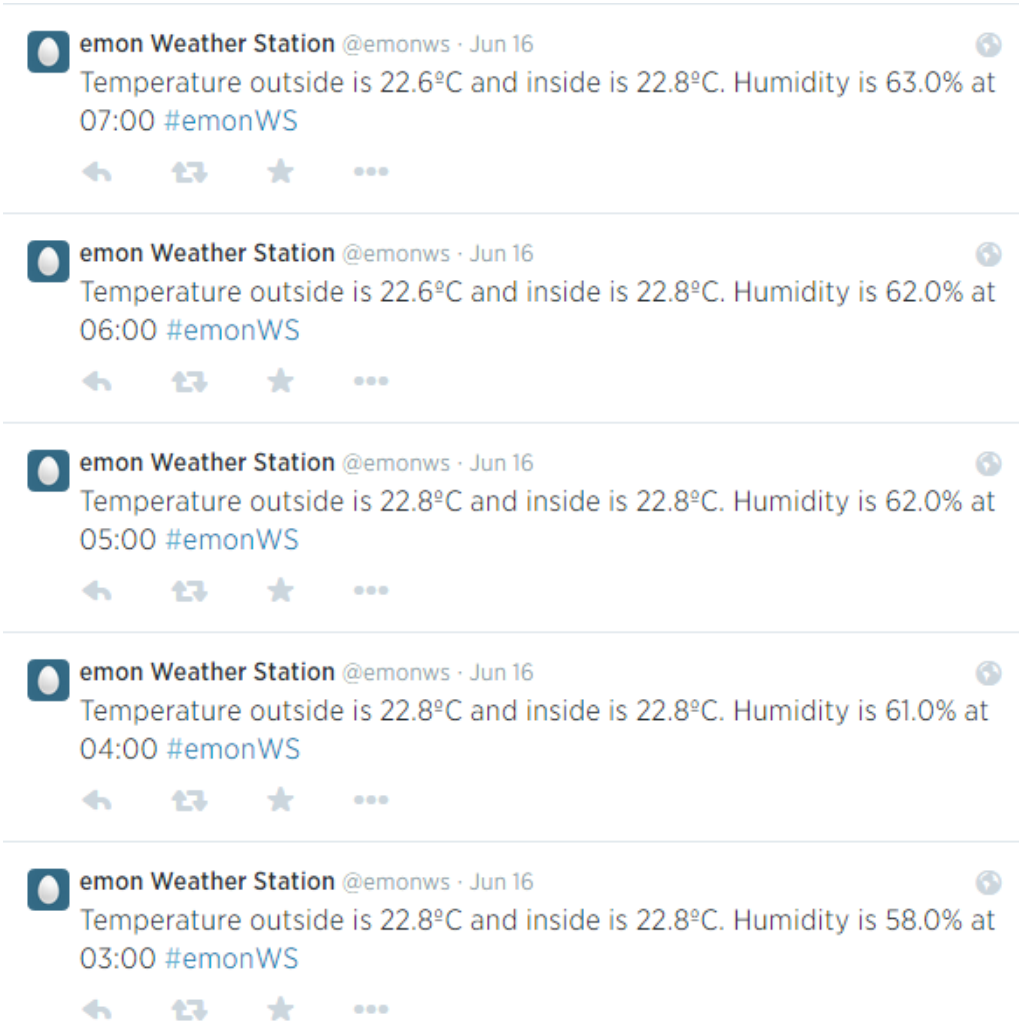
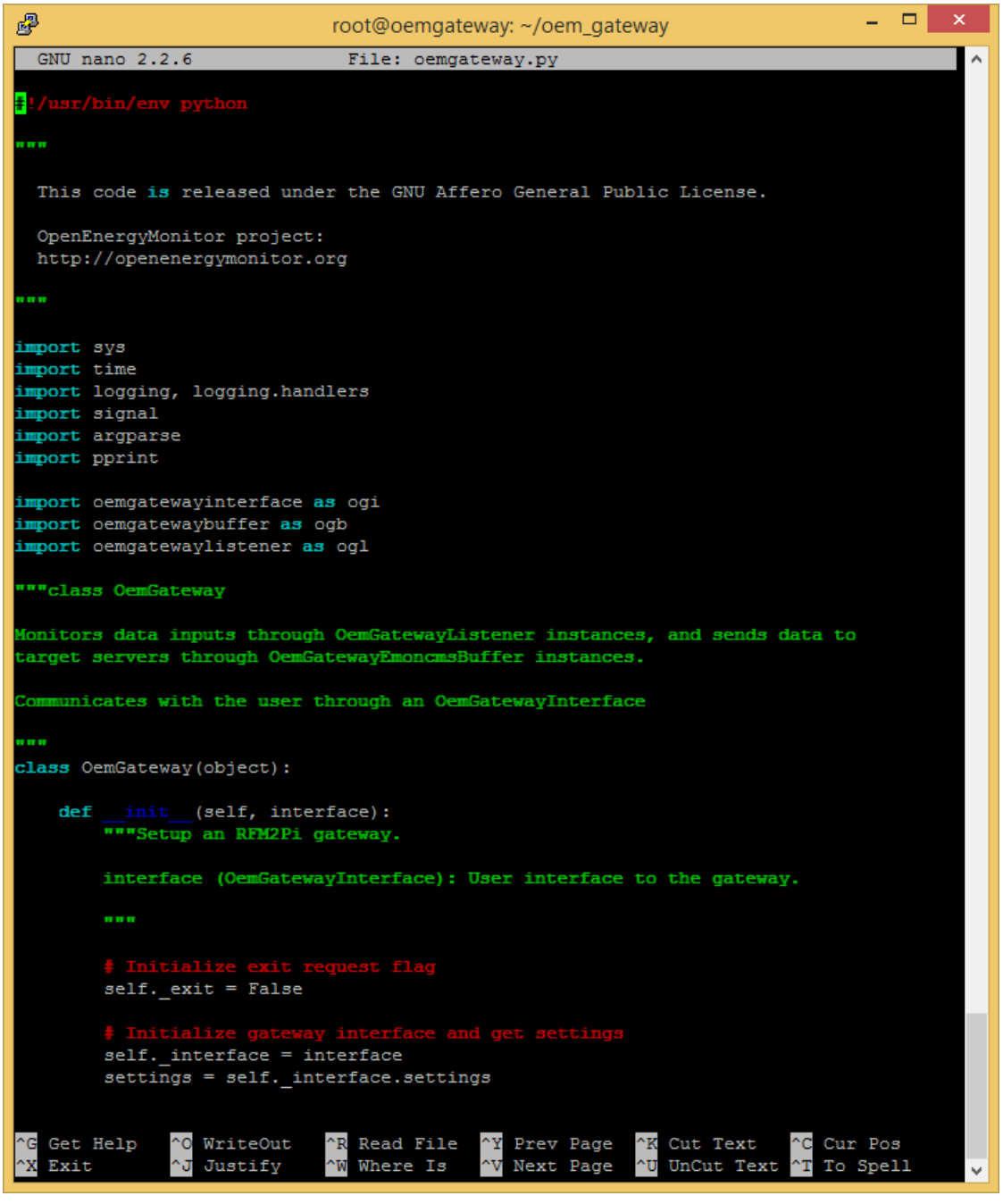


Figura. Captura con diversos tweets.

El cliente de Twitter usado es “ttytter”, un cliente de consola. Es un cliente liviano y capaz de realizar todo tipo de acciones con una simple línea de comandos. El uso del cliente no es muy intuitivo pero gracias a una serie de “alias” se pueden conseguir resultados muy satisfactorios.

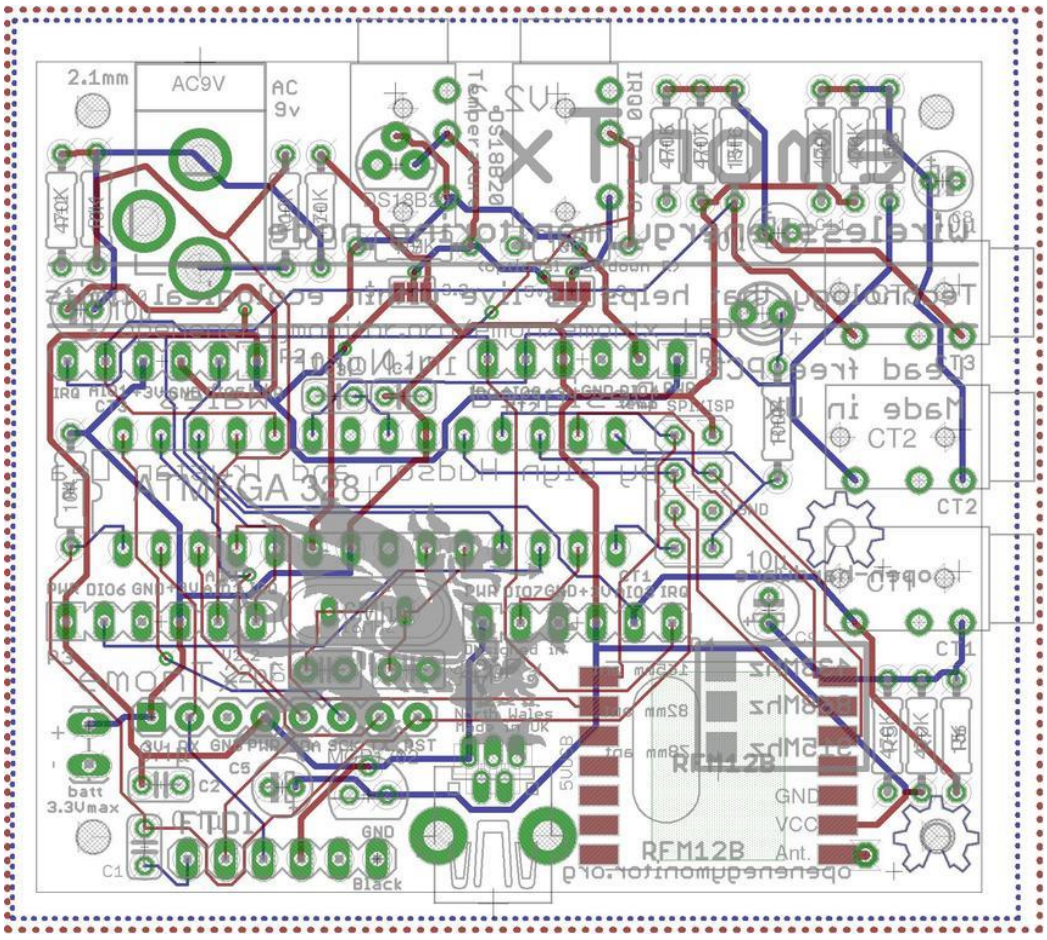
11.3. Scripts de comunicación



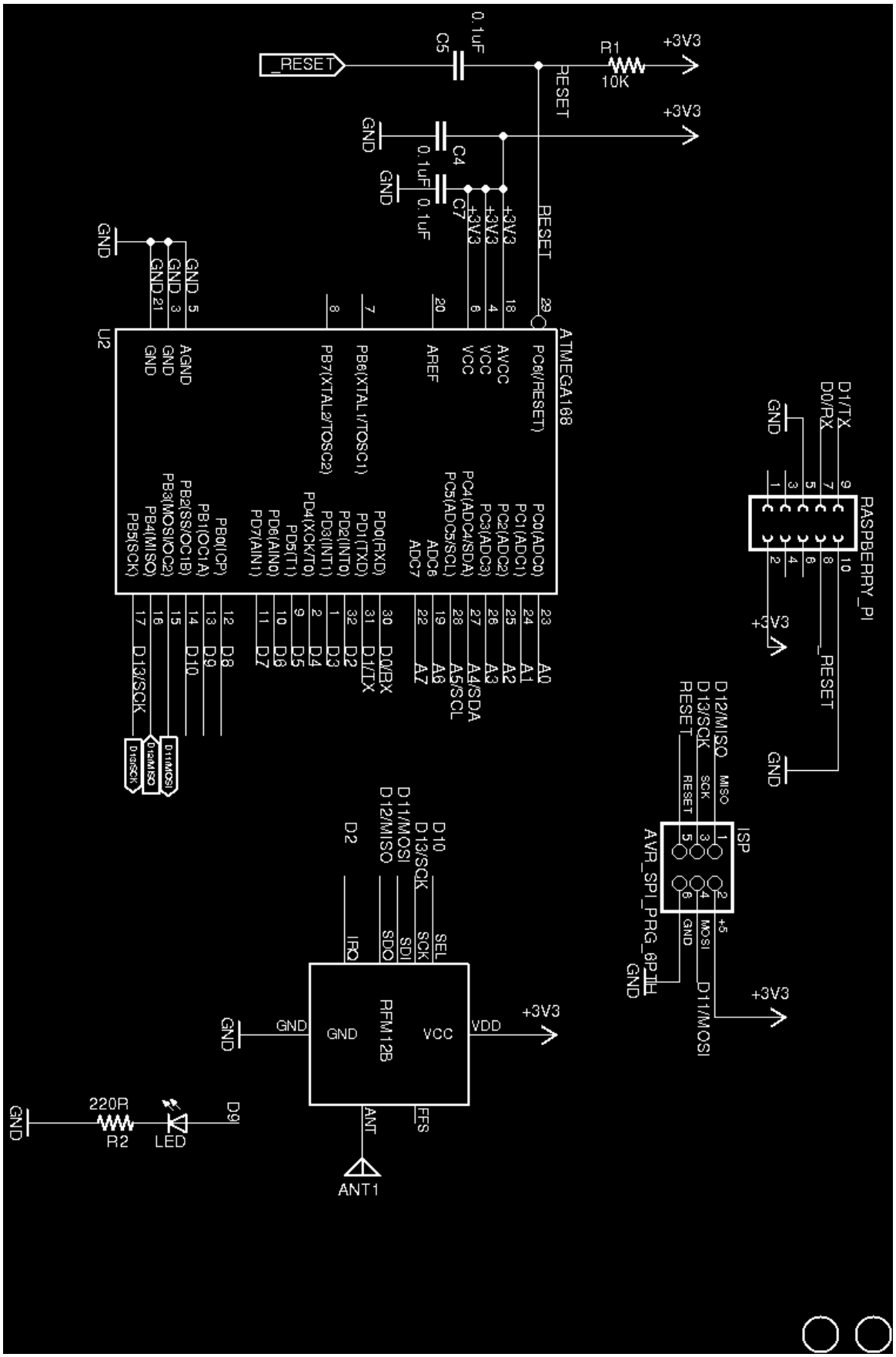
Fragmento del script necesario para la comunicación entre dispositivos

12. Anexo V – Datasheets e información técnica

12.1. Esquemáticos



Esquemático del nodo emonTx



Esquemático del módulo *RFM12Pi* conectado a la Raspberry Pi