E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

Monitorización del consumo eléctrico de un hogar: Procesado de datos mediante Arduino

Grado en Ingeniería
en Tecnologías Industriales

Trabajo Fin de Grado

Víctor Erice Carbonero

Tutor: Dr. Jesús López Taberna

Cotutor: Dr. Javier Marcos Álvarez

Pamplona, 30 Junio 2015

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

AGRADECIMIENTOS

A mi familia y amistades.

A mis compañeros, con especial mención a mis compañeros de proyecto.

A mis tutores Dr. Jesús López Taberna y Dr. Javier Marcos Álvarez y el doctorando Alberto Berrueta Irigoyen.

ABSTRACT

Nowadays, different types of smart meter for electric consumption are being developed. They allow the user to know the electric consumption at home at any time through the representation of the data in different graphics. The reason of this fast development is the increasing need of knowledge from the electricity generation enterprises. It is also a useful tool from the consumer's point of view, since it allows them to apply electricity saving measures.

This project has the aim to develop a smart meter. To do so, the work was distributed in different parts. This memory is focused in the signal processing and the data upload to a spreadsheet in Google Drive by using Arduino microcontroller.

An Arduino Mini is used to sense the current every second, the measurement will be sent each second and a half by the module FS1000A that is a radiofrequency transmitter to the Arduino Uno which is connected to Internet through an Ethernet cable. The Arduino Uno will receive data through the receiver module XY-MK-5V and, upload them in the Cloud every 10 minutes. The electricity consumption will be stored in the Cloud in a Google Drive spreadsheet, a form will be used in order to add each new datum.

KEYWORDS

- Smart meter
- Arduino
- Google Drive
- Ethernet
- Radiofrequency

RESUMEN

En el mercado actual se están desarrollando diferentes tipos de contadores de consumo eléctrico inteligentes, los cuales permiten al usuario conocer en cualquier instante el consumo de electricidad del hogar mediante la representación de los datos medidos en diferentes gráficas. Esto es debido a la necesidad creciente de un conocimiento mayor del estado del consumo por parte de las empresas comercializadoras de electricidad. Por parte de los consumidores también se ha creado la necesidad del conocimiento instantáneo y a lo largo del tiempo, ya que facilita implementar medidas de ahorro energético.

Este proyecto consistirá en desarrollar un contador de consumo eléctrico inteligente, para ello se distribuirá el trabajo a realizar en diversas partes. Esta memoria se centrará en el procesado de la señal y la subida de datos a una hoja de cálculo de Google Drive mediante las herramientas que facilita la plataforma Arduino.

Se empleará un Arduino Mini para realzar el sensado cada milisegundo, enviará cada segundo y medio mediante el módulo FS1000A, emisor de radiofrecuencia, un dato a un Arduino Uno conectado a una red de internet mediante cable Ethernet. El Arduino Uno recibirá el dato a través del módulo recepto XY-MK-5V el dato y lo subirá a la Nube cada 10 minutos. En la Nube se almacenará mediante una hoja de cálculo, para introducir los datos se empleará un formulario.

PALABRAS CLAVES

- Contador inteligente
- Arduino
- Google Drive
- Ethernet
- Radiofrecuencia

ÍNDICE

ÍNDICE FIGURAS

ÍNDICE TABLAS

1 INTRODUCCIÓN

En el mercado actual se están desarrollando diferentes tipos de contadores de consumo eléctrico inteligentes, debido a la necesidad creciente de un conocimiento mayor del estado del consumo por parte de las empresas comercializadoras de electricidad, como se ha podido comprobar recientemente en el Estado español, y por otro lado, por parte de los consumidores también se ha creado la necesidad del conocimiento instantáneo y a lo largo del tiempo, ya que facilita implementar medidas de ahorro energético.

También desde un enfoque a nivel de investigación, resulta interesante implementar este tipo de contadores, debido a que de una forma sencilla, tendremos almacenados los datos medidos automáticamente, y con fácil acceso a ellos.

Por otro lado, el desarrollo actual de un nuevo tipo de configuración de la red, conocido como Smart Grid, implica la instalación de este tipo de contadores para los consumidores, ya que como se ha comentado previamente, existe una necesidad del conocimiento de la red, en la zona de consumo.

Por todo esto, se ha visto que hay un interés en desarrollar un contador de consumo eléctrico inteligente.

2 OBJETIVO

Se pretende diseñar y construir un contador de consumo eléctrico inteligente que cumpla ciertas condiciones, para que resulte útil, tanto de manera comercial, para el uso en viviendas, como para desarrollar investigaciones.

Para ello se exigirá que el contador de consumo eléctrico inteligente a desarrollar cumpla los siguientes requisitos:

- Bajo precio.
- Medidas similares a un contador comercial.
- Lectura de los datos desde internet.
- Fácil instalación.

3 ESTADO ACTUAL

A continuación se estudiará el estado actual de los contadores de consumo eléctrico inteligentes, que se pueden encontrar comercialmente, ya que de esta manera, se tendrá una idea del estado actual del mercado y las diferentes alternativas que presentan los contadores comerciales y proyectos paralelos que se han ido desarrollando de libre acceso.

En el mercado actual, nos encontramos principalmente con tres alternativas, mediantes las cuales podemos ver la evolución que han tenido los contadores de consumo eléctrico inteligentes, las cuales son:

1. **Visualización de datos mediante pantalla LCD.**

Este modelo destaca por su simplicidad, ya que únicamente es un contador de consumo eléctrico el cual incorpora una pantalla LCD, lo que resulta atractivo para personas que carecen de conocimientos tecnológicos. También está pensado para personas que no buscan datos específicos ni una base de datos amplia de estos, es decir que tengan un interés por el estado instantáneo de consumo y una comparativa de cómo máximo el mes actual con el mes anterior.

Figura 1: Owl Micro + [1]

Otra de sus características es que incorpora una alarma, la cual te puede avisar en diferentes tipos de situaciones, como sobrepasar el límite de potencia instantánea consumida o cierto valor de energía consumida.

La última característica destacable, es la comunicación por radiofrecuencia entre el sensor y la pantalla LCD.

2. Visualización de datos mediante pantalla LCD + exportación de datos a un ordenador (USB).

Esta opción, es muy similar a la anterior, pero aporta la posibilidad de exportar los datos mediante USB, por lo que tendremos una mayor capacidad de almacenamiento de datos, pero puede resultar bastante incomoda si requieres trabajar con datos instantáneos.

Figura 2: Current cost EnviR [2]

Tanto este tipo de contador, como el anterior, no resultan útiles como herramienta en el campo de la investigación, debido a lo comentado anteriormente. Pero por otro lado como dispositivo comercial, sí que tiene gran utilidad, ya que son dispositivos sencillos.

3. Exportación de datos a una Nube y visualización vía página web.

Esta última opción, en vez de tener la posibilidad de visualizar los valores en un único dispositivo y para poder acceder a la base de datos tener que exportar los datos, lo hace el contador automáticamente, almacenando los datos en una "Nube", es decir en un servidor de internet.

Esto posibilita el acceso a una gran variedad de datos, y la posibilidad de trabajar con ellos de diferentes maneras. Y también el poder tener acceso a los datos desde cualquier punto de la red.

Figura 3: Engage hub kit [3]

Dentro de este tipo de contadores de consumo se puede encontrar una gran variedad de formas de visualización de los datos en la red. Se analizara a continuación dos alternativas:

- La alternativa propuesta por Engage Efergy, como se observa en la figura 4, da la posibilidad de ver el consumo en diferentes periodos, pero no te ofrece una comparativa dentro de un mismo periodo. Por otro lado podemos ver en términos tanto de energía como monetario lo que se ha ido consumiendo a lo largo del día, la semana o el mes. Por ultimo cabe destacar que te da la opción de descargar los datos.

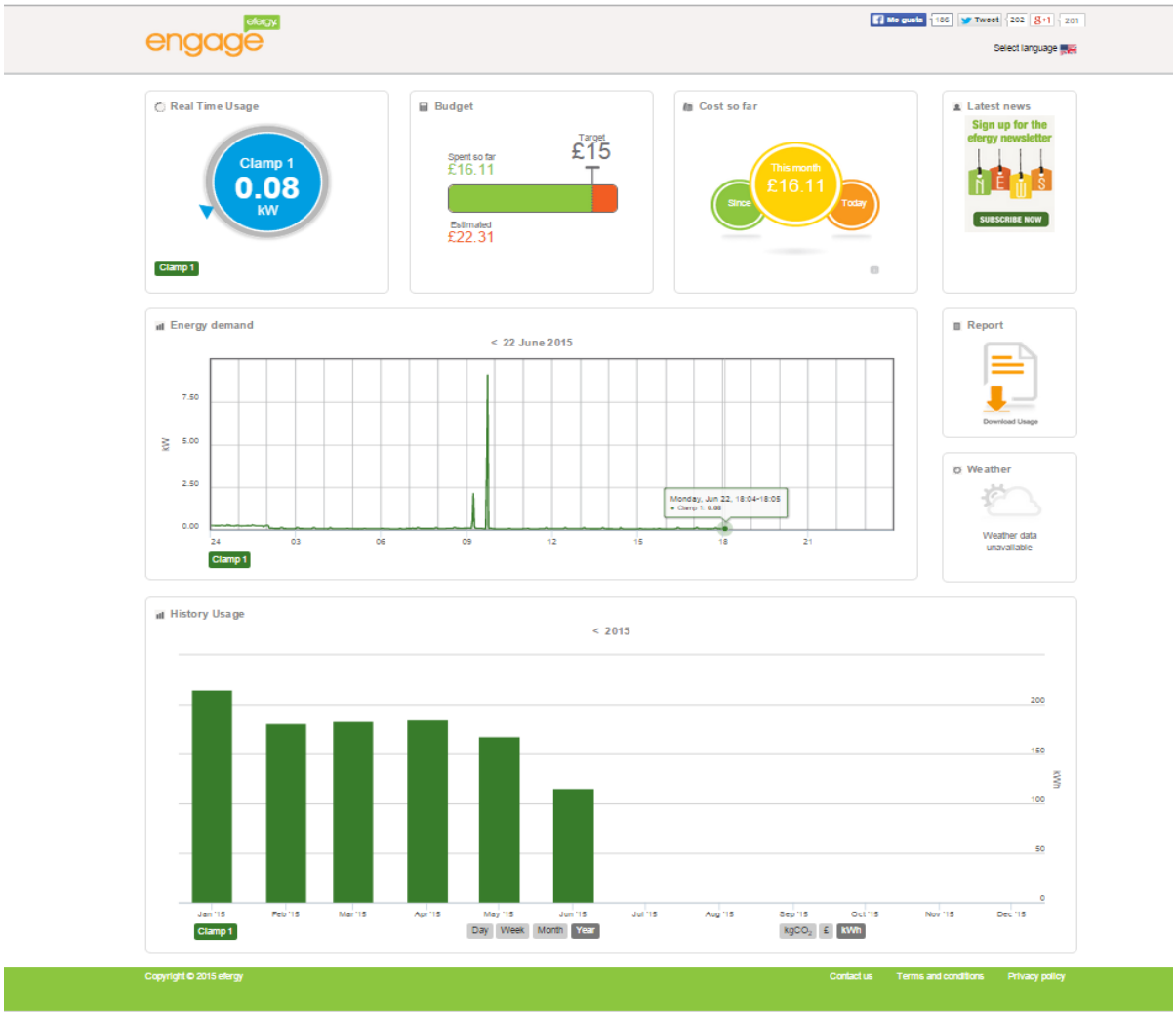


Figura 4: Pagina web de Engage Efergy [4]

- La otra alternativa analizada es la propuesta por la empresa Current Cost. Este modelo es más simple que el anterior, ya que puedes observar de la evolución de la potencia consumida, pero en periodos como máximo mensuales. Aunque te ofrece la posibilidad de descarga de datos de hasta dos años.

Figura 5: Pagina web de Current Cost [5]

4 ORGANIZACIÓN DEL PROYECTO

4.1 Distribución del proyecto

Tras analizar diferentes alternativas, se tiene que definir las características del contador que se ha de realizar, para ellos y debido a las condiciones externas que se definen, el proyecto se divide en tres partes diferentes:

4.1.1 Sensado y acondicionamiento

En la primera parte proyecto, se desarrollará la parte del sensado y acondicionamiento de la señal producida por el sensor, para adaptarla a los dispositivos que se emplearán para procesarla.

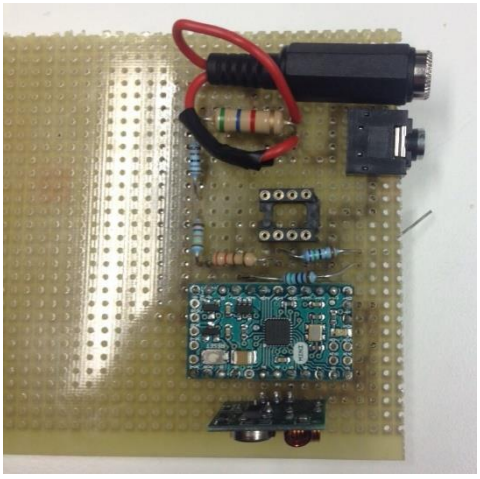


Figura 6: Acondicionamiento de la Señal, Arduino Mini y módulo FS1000A

De esta etapa, se encargará mi compañero Joseba Revuelta Irrisari. [6]

4.1.2 Procesado y subida de datos

La siguiente parte en la que se descompondrá el proyecto, consistirá en el procesamiento de los datos sensados y su posterior almacenamiento en la “Nube”.

Para ellos se desarrollaran dos partes paralelas, una de ellas empleando un microcontrolador, basándose en la plataforma de hardware libre Arduino, y por otro lado se empleará una Raspberry PI, es decir, un ordenador de placa reducida.

El proyecto a desarrollar empleando un microcontrolador se encargará el redactor de la presente (esta) memoria. Y del proyecto empleando la Raspberry Pi se encargará Aritz Legarrea. [7]

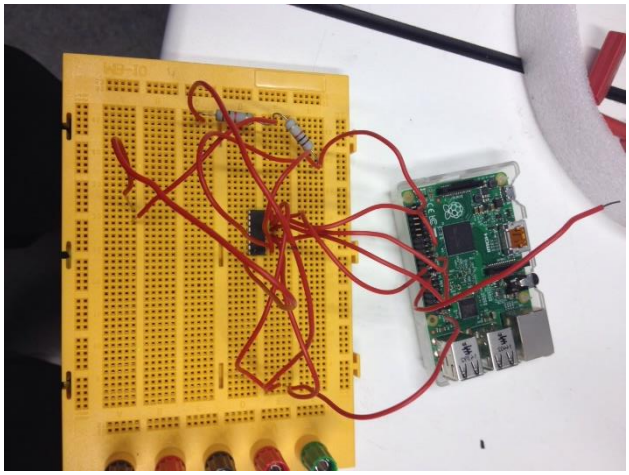


Figura 7: Acondicionamiento de la señal y Raspberry Pi 2.b

4.1.3 Diseño de la página web

En la última parte, en la cual se divide el diseño del contador, se diseñará una página web para la visualización e interactuación de los datos obtenidos. Se empleará la plataforma Shiny, la cual emplea el lenguaje de programación R y permite de forma sencilla realizar la programación de una página web.

Figura 8: Inicio de la página web

De esta última parte se encargará mi compañera Naia Barriola Hernandorena. [8]

4.2 Decisiones generales

Tras la distribución del proyecto, se procedió a tomar una serie de decisiones, para dotar al contador de consumo eléctrico inteligente de una serie características, y de las herramientas que íbamos a emplear para conseguir estas características en él.

Como ya se ha comentado, ciertas herramientas y características se definieron en el planteamiento de la propuesta de:

- El empleo de al menos una de las placas que proporciona la plataforma Arduino, por lo que posteriormente se deberá seleccionar cual o cuales emplear.
- El empleo de la última versión que se ha comercializado de Raspberry Pi.
- El empleo de la plataforma Shiny para la programación y diseño de la página web.

El equipo de trabajo tomó las siguientes decisiones para delimitar y definir los requerimientos técnicos que debía tener el contador.

Todas las decisiones iniciales que atañen únicamente al Arduino se trataran posteriormente.

4.2.1 Sistema de almacenamiento en la Nube

La primera decisión que se considera es qué sistema de almacenamiento en la Nube se va a emplear.

En internet, se puede encontrar con diferentes plataformas y servidores, que tienen como único fin el que los usuarios de estas plataformas puedan almacenar diferentes archivos en ellas. A este tipo de páginas webs, se les denomina comúnmente sistemas de almacenamiento en la Nube.

Se tiene la necesidad de emplear este servicio, a causa de que la otra alternativa era crear nuestro propio servidor, como lo hacen la mayoría de las empresas comercializadoras de este tipo de contadores. Pero esta alternativa no es viable, a causa de la gran complejidad que conlleva crear un servidor.

Para ello se analizaron una serie de plataformas:

- Dropbox: plataforma con una capacidad de almacenamiento inicial de 2 GB, los cuales se pueden amplificar de forma gratuita cumpliendo ciertas

condiciones o con diversos planes de pago. Permite la posibilidad de compartir archivos de forma pública o privada. Como principal desventaja difícil sincronización entre varias carpetas. [9]

Figura 9: Logotipo Dropbox [9]

- Google Drive: plataforma con una capacidad de almacenamiento inicial de 15 GB, la única forma de ampliar la capacidad es pagando. Como ventaja principal, es que posee integrado el antiguo servicio de Google Docs, lo que nos permite crear diferente tipo de documentos, hojas de cálculo y similares desde la misma plataforma. Estos archivos creados desde Google Drive no computaran en la capacidad de almacenamiento. [10]

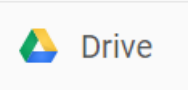


Figura 10: Logotipo Google Drive [10]

- OneDrive: plataforma oficial de Microsoft, tiene una capacidad de almacenamiento inicial de 15 GB, con posibilidad de aumentarla mediante pago. Como Google Drive tiene integrado un servicio de para la creación y trabajo de documentos online, lo que puede ser una ventaja, pero al no ser programas abiertos, solo puedes trabajar con los tipos de archivos oficiales de Microsoft. [11]

Figura 11: Logotipo OneDrive [11]

El requerimiento principal que se impone al sistema de almacenamiento en la Nube, con el que se va a trabajar, es que nos permita trabajar con una hoja de cálculo o base de datos online.

Podría haber un requisito de memoria, ya que el archivo conforme vaya pasando el tiempo ocupará mayor espacio de memoria. Pero en diversas alternativas no ocupará espacio de memoria, ya que se emplea un archivo creado desde el sistema.

	Dropbox	Google Drive	OneDrive
Memoria Gratuita	2 GB	15 GB	15 GB
Capacidad de trabajo con documentos online	No	Si	Si
Ventaja	Simplicidad de uso	Empleo de software libres, documento creados desde Google Docs no ocupan espacio	Gran capacidad gratuita
Desventaja	Solo se puede sincronizar una carpeta	Necesidad de cuenta de Gmail para crearlo	Limitación ya que solo se pueden emplear productos oficiales de Microsoft

Tabla 1: Comparativa de Sistemas de almacenamiento en la Nube

Por todo esto, se puede concluir que el sistema de almacenamiento en la "Nube" que mejor se ajusta a los requisitos requeridos es Google Drive. Ya que este nos proporciona la posibilidad de trabajar con documentos online, creados desde la plataforma, y estos no ocuparán espacio en la memoria, como se observa en la tabla 1.

Para el almacenamiento de datos se planteó en dos tipos diferentes de archivos, una hoja de cálculo o una base de datos. Por simplicidad y conocimientos previos del grupo de trabajo, se decidió emplear una hoja de cálculo.

4.2.2 Sensado

La siguiente decisión a tomar es qué sensar, para ello se propone sensar con prioridad la intensidad, y posteriormente se podrá estudiar la posibilidad de sensar la tensión.

Se ha tomado esta decisión debido a que no se ha considerado que el sensar la tensión fuese fundamental, ya que la tensión presenta pequeñas fluctuaciones en torno a 230 V. Medir la tensión solo tiene cierto interés cuando se quiere conocer el factor de potencia, lo que actualmente solo resulta interesante para aplicaciones en el campo de la investigación.

4.2.3 Conexiones a internet

La siguiente decisión que se trata, es la manera de conectar los dispositivos. Por simplicidad se decide que a la Raspberry Pi, se le conectase una antena WiFi, ya que no se incrementa de una manera desmesurada el precio. El tipo de conexión que empleará el Arduino se tratará posteriormente.

4.2.5 Tiempo de muestreo

El tiempo de muestreo en ambos casos de la lectura de datos se decide que se aproxime a 1 milisegundo, ya que se considera que de esta manera se obtendrá una precisión óptima para el contador.

4.2.6 Página web

Otra de las decisiones fundamentales a tratar es que se quiere mostrar en la página web, para ello lo primero que se tiene que determinar es cada cuanto se actualizará el valor instantáneo, es decir, por hacer una analogía, el periodo de muestreo de la página.

Se decide buscar un término medio entre requerimientos que podría exigir un contador centrado en el campo de la investigación o un contador centrado en una aplicación comercial para viviendas, por lo que se muestreará cada 10 minutos.

A partir de aquí, teniendo delimitado el tiempo mínimo, se concluye que se realizarán unas graficas:

- Diarias
- Mensuales
- Anuales

Por otro lado también se propone mostrar momentos de máximo mínimo consumo, valor instantáneo de potencia, factura de la luz aproximada y diversos consejos para ahorro de consumo eléctrico.

5 PROCESADO Y SUBIDA DE DATOS CON ARDUINO

5.1 Fundamentos de Arduino

Antes de empezar a desarrollar esta parte de este proyecto, se ha de conocer con que se va a trabajar, por lo que se va analizar lo qué es un Arduino.

Figura 12: Logotipo Arduino [12]

El concepto de Arduino abarca tres conceptos.

5.1.1 Placa de hardware libre

Es una placa de hardware (PCB) libre que incorpora un microcontrolador reprogramable y una serie de pines (hembra), los cuales se unen internamente al microcontrolador. Estos pines permiten conectar de forma sencilla diferentes componente.

Arduino tiene diferentes modelos, por lo que se ha de especificar sobre qué modelo estas trabajando, los cuales tienen diferentes características. Esto hace tener varios microcontroladores, de la misma familia tecnológica, ya integrados en PCBs. Lo que simplifica el empleo de estos, ya que no se requiere acondicionar el microcontrolador.

Los microcontroladores que emplea Arduino son desarrollados y construidos por Atmel, la gran mayoría emplean una arquitectura AVR, salvo el microcontrolador del Arduino Due.

El que sea un hardware libre posibilita el poder entender su funcionamiento, modificarlo, mejorarlo y compartir dichas modificaciones y mejoras. Esto además lo facilita la compañía de Arduino, ya que hacen públicos los esquemas del diseño del hardware de todos sus modelos. Todo esto conlleva a que sea muy fácil encontrar diferentes modelos no oficiales, que a partir de un modelo oficial se ajustan más a unos requerimientos específicos.

5.1.2 Software libre

Arduino también incorpora un software gratis y libre. El cual puede funcionar con varios sistemas operativos, que se instala en el ordenador desde el cual se va a programar la placa de Arduino.

Este software permite escribir, compilar y cargar en la memoria del microcontrolador un conjunto de instrucciones, que posteriormente se ejecutaran desde el microcontrolador. Es decir nos ofrece una IDE gratuita para la programación del microcontrolador.

Este software permite al hardware del Arduino a trabajar sin tener la necesidad de estar conectado a un ordenador, es decir que funcionará de forma autónoma si dispone de una fuente de alimentación.

Al ser un software libre se tienen las siguientes ventajas:

- Libertad de usar el programa con cualquier propósito y en cualquier sistema informático.
- Libertad de estudiar el funcionamiento interno del programa y adaptarlo a las necesidades particulares.
- Libertad para mejorar el programa y hacer públicas las mejores. Por lo que se puede encontrar modificaciones del software, que se adapte mejor a las necesidades particulares.

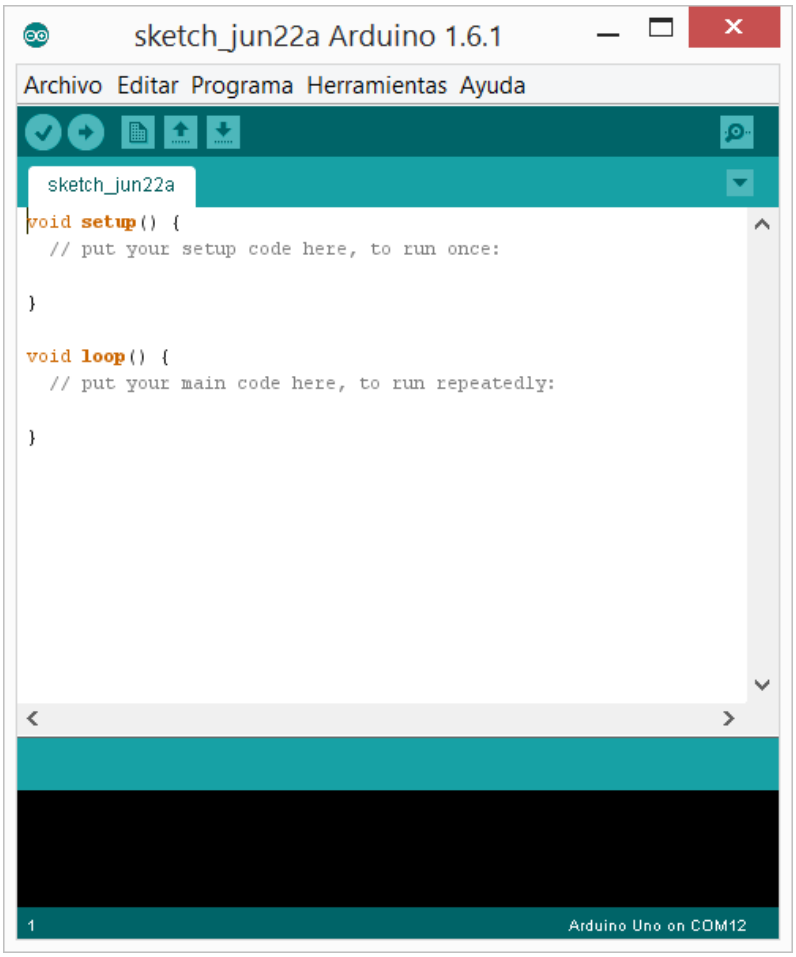


Figura 13: IDE Arduino

En la IDE de Arduino se suele estructurar el programa de la siguiente manera:

1. Definición de constantes, variables globales, librerías y programas que se van a emplear a lo largo del programa que se va a desarrollar.
2. Void setup(), en él se realizan las acciones que únicamente es necesario ejecutar una única vez.
3. Void loop(), en esta pare se introducen las acciones que se quiere que se repitan permanentemente.
4. Void nombre_funcion(), cuando quieres definir una función que luego se va a integrar en el setuo o loop, se realizan de esta manera para facilitar la comprensión del código o cuando se va a repetir varias veces la función a lo largo del programa.

5.1.3 Lenguaje de programación

Arduino ha desarrollado un lenguaje de programación libre basado en C/C++, por lo que incorpora funciones específicas, que hacen la programación de este muy sencilla, además posibilita que las personas desarrollen diversas funciones nuevas y que se puedan emplear mediante librerías.

Estas tres características que definen lo que significa Arduino hacen que el empleo de este resulte atractivo, ya que no tienes restricciones a la hora de poder hacer modificaciones, hay una gran variedad de modelos y se puede acceder información sobre ellos y aplicaciones que se pueden desarrollar con ellos.

5.1.4 Arduino Shields

Además Arduino y diferentes entidades han desarrollado complementos para los diferentes Arduinos, estos complementos son conocidos como Shields. Estas Shields suelen ser PCBs que dotan a los Arduinos de nuevas características que nuestro

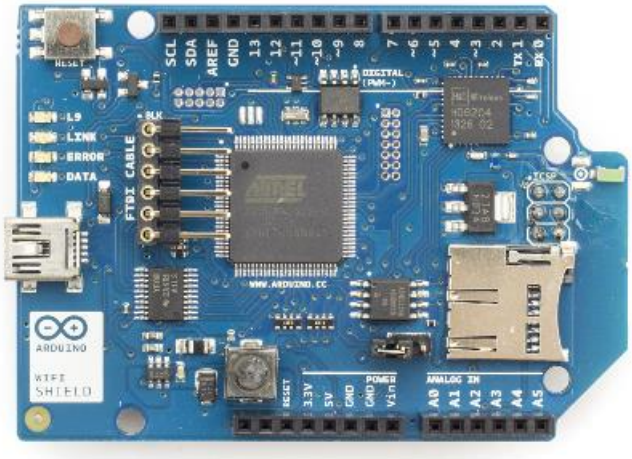


Figura 14: Arduino WiFi Shield [12]

Arduino no tiene, como por ejemplo la Arduino Ethernet Shield, la cual dota al Arduino de poder conectarse a internet vía Ethernet.

Estas Shields también tienen el hardware libre, por lo que como cualquier modelo de Arduino se pueden encontrar diversas modificaciones y versiones que se pueden ajustar mejor a las características que se desean.

5.2 Elección de Arduino

La siguiente parte a desarrollar en el trabajo correspondiente a esta memoria fue la elección del tipo de Arduino a emplear.

Para ello lo primero que se hizo fue estudiar los requerimientos fundamentales que se tenían en esta situación, los cuales son los siguientes:

- Capacidad de tener acceso a internet.
- Capacidad de muestreo cada milisegundo.
- Una entrada analógica.
- No se requiere una memoria extensa.

Como se puede ver el requisito más restrictivo es el tener capacidad de acceso a internet, pero como ya se ha mencionado previamente, existe la posibilidad de emplear una Arduino Shield que nos permita esta conexión.

Así que inicialmente se analizaron las siguientes posibilidades:

5.2.1 Alternativa 1

Esta alternativa se compondría principalmente de dos elementos:

1. Arduino Uno: este Arduino basado en el microcontrolador ATmega328, contiene 14 entradas o salidas digitales, 6 entradas analógicas, una velocidad de reloj de 16 MHz y una memoria flash de 32 kB. Se puede alimentar en un rango recomendado entre 7 y 12 V, pudiendo soportar hasta 20V. Como se ha comentado anteriormente este modelo de Arduino

Figura 15: Arduino Uno [12]

cumpliría con todos los requisitos, a excepción de la capacidad de conectarse a internet. Lo que conlleva el siguiente elemento.

2. Arduino Ethernet Shield: esta Shield nos permite conectarnos a internet, via Ethernet, es decir dota de esta capacidad a los Arduinos compatibles con este. En el caso de Arduino Uno resultan compatibles, inhabilitando 4 patillas digitales del microcontrolador.

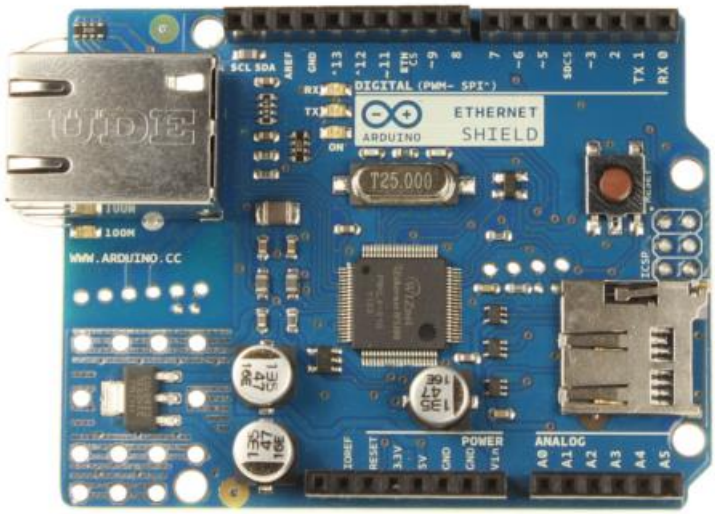


Figura 16: Arduino Ethernet [12]

Esta solución tiene el inconveniente de que se necesitara proximidad entre el acceso a Ethernet y el punto a sensar.

Por otro lado el precio, obtenido desde rs-online.com, de esta combinación es de:

- Arduino Uno=21.10 €
- Arduino Ethernet Shield=31.57 €
- Total= 52.67 €

5.2.2 Alternativa 2

Esta posible combinación es similar a la anterior, pero en vez de emplear una Arduino Ethernet Shield, se empleará una Arduino WiFi Shield.

Esta shield dota también a un Arduino Uno de conexión a internet, pero como se ve reflejado en sus nombres, en esta ocasión se conectará a internet vía WiFi. Esto quita la limitación que provoca el empleo de la anterior shield, ya que de esta manera no se tendrá la necesidad de tener próximo el punto de sensado y la conexión del Ethernet, o tener que emplear un cable Ethernet de gran longitud.

El precio, obtenido desde rs-online.com, de esta alternativa queda de la siguiente manera:

- Arduino Uno=21.10 €
- Arduino WiFi Shield=80,19 €
- Total=101.29 €

5.2.3 Alternativa 3

Esta alternativa se basa únicamente en el empleo de un Arduino Yún.

El Arduino Yún es una placa basada en el microcontrolador ATmega32u4 y el Atheros AR9331, lo que dota a este Arduino de un procesador Linux de apoyo. Además la misma placa incorpora un soporte que te permite conectarte tanto vía Ethernet como vía WiFi. Por otro lado tiene 20 pines digitales, de los cuales 12 se pueden utilizar como entradas analógicas, pose una velocidad de reloj de 16 MHz.

Figura 17: Arduino Yún [12]

Esta opción integra todos los requerimientos que se le exige en una sola placa. Su precio, obtenido desde rs-online.com, es de:

1. Arduino Yún=71,36 €

5.2.4 Alternativa 4

Esta alternativa se desarrolla a partir de la primera, intentado eliminar el inconveniente de esta, ya que la alternativa 1 está limitada por la longitud del cable Ethernet. La solución que se emplea se basa en una de las soluciones comerciales de los contadores de consumo, la cual es emplear comunicación de radiofrecuencia entre la parte del sensado y la parte encargado de subir los datos. Por lo que se decide emplear un Arduino más que se situará en la parte de sensado, el cual no tendrá la necesidad de tener conexión a internet y otro cerca del acceso a Ethernet el cual tendrá los requerimientos anteriores.

Esta alternativa se compone de los siguientes elementos:

1. Arduino Mini: es una pequeña placa basada en el microcontrolador ATmega168, mantiene la velocidad de reloj de los anteriores Arduinos. Posee menor número de entradas tanto analógicas como entradas y salidas digitales. Se alimenta a una tensión de entre 7 y 9 V.

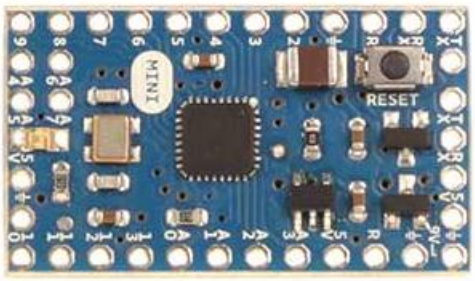


Figura 18: Arduino Mini [12]

2. Módulo *FS1000A* (transmisor) + Módulo *XD-RF*-5V (receptor), son dos módulos que están preparados para comunicarse entre ellos, en un solo sentido, mediante una señal de radiofrecuencia a 433MHz. Su tensión de trabajo es de 5 V, lo cual facilita integrarse con los Arduinos que se emplean en esta alternativa, ya que los dos tienen una salida a 5 V.

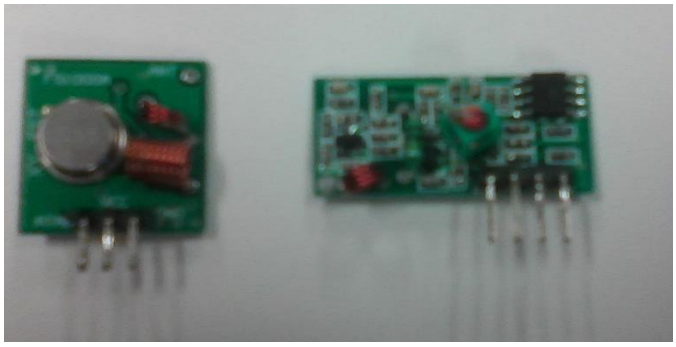


Figura 19: Módulo FS1000A (izquierda) y Módulo XY-MK-5V (derecha)

3. Arduino Uno + Arduino Ethernet Shield: se estudió en el apartado anterior.

El precio, obtenido desde rs-online.com, de esta alternativa será descompondrá en:

- Arduino Mini=14.19 €
- Conjunto Radiofrecuencia= 2.32 € (obtenido en dx.com)
- Arduino Uno=21.10 €
- Arduino Ethernet Shield=31.57 €
- Total= 69.18 €

Como se puede observar en la tabla 2, la opción más barata seria la alternativa 1, pero debido a su limitación ya mencionada, se decidió decantarse por la última alternativa, ya que resultaba la más interesante de implementar.

	Alternativa 1	Alternativa 2	Alternativa 3	Alternativa 4	Referencia
Arduino Uno	21.1	21.1	-	21.1	A000073
Arduino Mini	-	-	-	14.19	A000088
Arduino Yún	-	-	71.36	-	A000003
Arduino Ethernet Shield	31.57	-	-	31.57	A000025
Arduino WiFi Shield	-	80.19	-	-	A000058
Modulos Radiofrecuencia	-	-	-	2.32	433Mhz RF Transmitter Module + Receiver Module Link Kit for Arduino / ARM /MCU WL - Green
Total	52.67	101.29	71.36	69.18	

Tabla 2: Precios Alternativas

Cabe mencionar que el precio del Arduino Yún al inicio del proyecto, cuando se tomó la decisión, es mayor que el precio actual, por lo que se podría contemplar actualmente emplear esta alternativa 3

5.3 Arduino Mini

A continuación se procede a programar el Arduino Mini, para ello, se tiene que tomar una serie de decisiones, ya que se debe tener claro qué trabajo tiene que realizar cada Arduino.

El primer aspecto que se analiza es si el Arduino Mini debe ser el único que trabaje con los datos o, por el contrario, que se reparta esta tarea con el Arduino Uno. Se ha optado por la segunda opción, para así liberar de una carga pesada al Arduino Mini. Además si se da el caso de que fallase la comunicación entre los Arduinos, el Arduino Uno no se quedaría sin datos, es decir que el Arduino Mini enviará varios datos al Arduino Uno cada diez minutos.

Por lo que se tiene que decidir cada cuánto va a enviar un dato, se toma la decisión de que envíe cada 1.5 segundos. De esta manera se asegura que en 10 minutos el Arduino uno reciba más de un dato, y de esta manera si se pierde un dato no afecte, de manera notoria, al valor medio de la I_{rms} de 10 minutos.

El segundo aspecto que se plantea, es como obtener el valor eficaz de la corriente. En este caso también se plantean dos alternativas:

- Primero sacar el valor máximo y posteriormente mediante la siguiente relación obtener el valor eficaz.

$$I_{RMS} = \frac{I_{max}}{\sqrt{2}} \quad (1)$$

Esta opción solo es correcta si tenemos una onda sinusoidal, es decir si se nos achata la señal no se puede emplear esta opción, pero al tener asegurada siempre una sinusoidal, no se tendrá ese problema.(referencia al trabajo de Joseba)

- Obtener el valor eficaz directamente mediante la siguiente expresión:

$$I_{RMS} = \sqrt{\frac{1}{T}\int_{t_0}^{t_0+T} i^2(t)dt} \qquad (2)$$

Esta opción, es la expresión general que define al valor eficaz.

Debido que se puede asegurar una onda sinusoidal, se puede optar por emplear la primera opción, a su vez, esta opción aporta simplicidad en el cálculo, lo que tiene por consecuencia, que el proceso sea más rápido y que haya menor probabilidad de retraso.

Previamente de realizar la programación se realiza el diagrama de flujo para tener claro el proceso a seguir, como se observa en la figura 20.

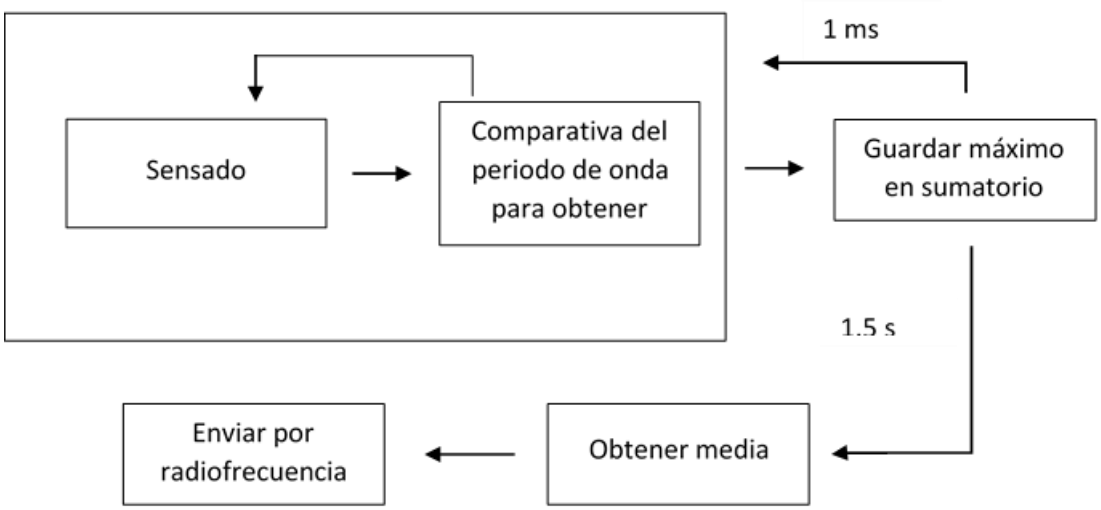


Figura 20: Diagrama de flujos de la programación del Arduino Mini

5.3.1 Lectura de datos

Tras la toma de decisiones anterior, se prosigue iniciando la programación del Arduino Mini, para ello se realiza una función, llamada sensado:

```
void sensado(){
tension_max_ciclo=0;
for (i=1;i<21;i++){
tension = analogRead(analog_pin);
if (tension>tension_max_ciclo){
tension_max_ciclo=tension;
```

```
}
delayMicroseconds(883);
}
}
```

Código 1

En esta función lo que se realiza es la obtención del máximo en un ciclo, como cada ciclo tiene un periodo de 20 milisegundos, se muestreará como se decidió, cada milisegundo. La operación de muestreo consiste en la lectura de la entrada analógica y comparación con el valor máximo, que al inicio de cada ciclo se reinicia a cero, y si este es mayor se guardará como valor máximo el valor que se acaba de obtener.

Como se considera que el Arduino tardará en realizar esa operación cierto tiempo, pero no un milisegundo, se tuvo que programar una espera de 883 microsegundos, como se verá en la prueba que se realizó.

Posteriormente se realiza los siguientes comandos dentro del loop:

```
sum_tension=0;
for (j=0;j< N_CICLOS;j++){
sensado();
sum_tension=sum_tension+tension_max_ciclo;
}
media_tension=sum_tension/ N_CICLOS;
```

Código 2

En estas líneas del *código 2* lo que se realiza es ejecutar cierto número de veces la función de sensado hasta llegar al segundo y medio y realizar la media de las tensiones máximas de cada ciclo.

Tras la realización de esta parte de la programación del Arduino Mini, se realiza una prueba para comprobar su correcto funcionamiento, por simplicidad, no se muestreará cada 1.5 segundos, sino que se empleará un paso menor de un segundo..

Tras la realización de una prueba se obtuvo que se retrasaba 117 milisegundos por segundo, por lo que se dedujo que en vez de tener un delay de un milisegundo se debía tener un delay de 883 microsegundos. Estos resultados se obtuvieron mediante la función milis, para así saber que retraso se obtenía cada segundo.

5.3.2 Emisor de radio frecuencia

La siguiente parte de la programación del Arduino Mini que se debe desarrollar es la emisión por radio frecuencia, para ello se emplea la librería VirtualWire [13].

Aparte de incluir la librería se ha de iniciar en el setup una serie de parámetros para la configuración correcta del emisor:

```
vw_set_ptt_inverted(true);
vw_setup(2000);
vw_set_tx_pin(3);
```

Código 3

La primera línea del *código 3* es algo necesario, ya que lo requiere la librería para activar, por así decirlo, el módulo de radiofrecuencia. La segunda línea nos define la velocidad de transmisión de bits que tendrá el emisor de radiofrecuencias. En la última línea se define el pin de salida que enviará el dato al emisor de radiofrecuencias.

Posteriormente en el loop habrá que incorporar:

```
str=String(media_tension);
 str.toCharArray(b,5);

 vw_send((uint8_t *)b, strlen(b));
 vw_wait_tx();
```

Código 4

La primera parte del *código 4* tiene el fin de convertir nuestro dato, que está definido como una variable numérica, a un dato definido como un carácter.

La segunda parte del *código 4* es la que envía el dato, y espera a que se haya terminado de enviar para poder seguir ejecutando el programa. De esta manera da tiempo a que se envíe completamente el dato, antes de enviar el siguiente dato.

Con esta parte ya se completa la programación del Arduino Mini, en el Anexo 1 se puede observar cómo quedaría la programación completa y la definición de sus variables.

5.4 Arduino Uno

Tras la realización de la programación del Arduino Mini, se ha de proseguir con la programación del Arduino Uno. Para ello, se ha de realiza un diagrama de bloques (figura 21), en el cual se esquematice las ideas principales del proceso que ha de realizar el Arduino Uno.

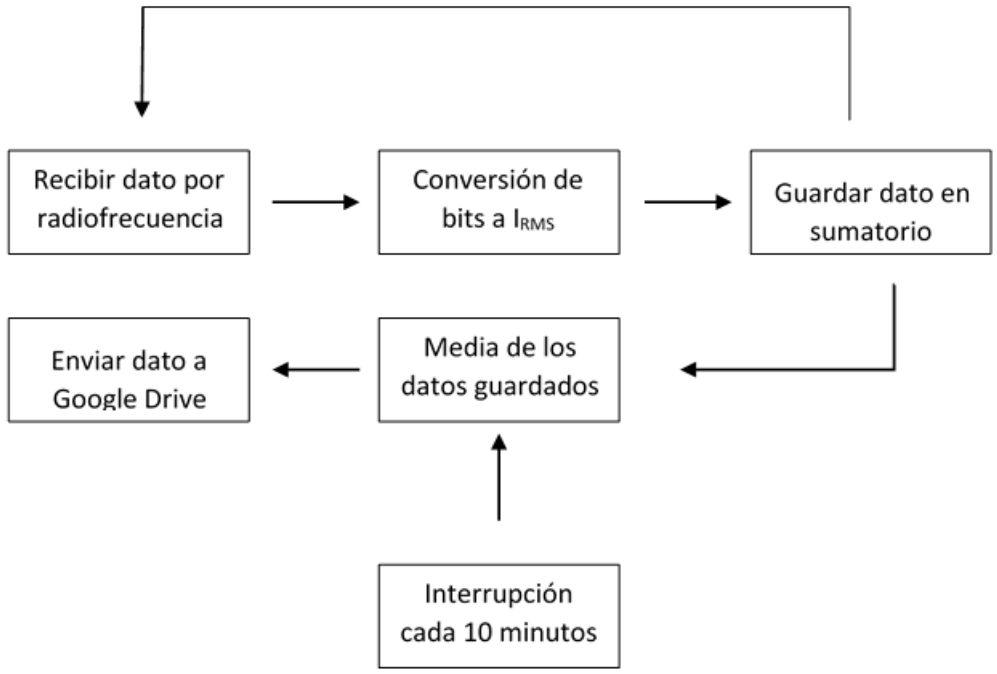


Figura 21: Diagrama de Flujo de la programación del Arduino Uno

5.4.1 Recepción de los datos

La primera parte que se ha de desarrollar es la recepción de datos, debido a que de esta manera se podrá comprobar el correcto funcionamiento de los módulos de radiofrecuencia y analizar hasta que distancia es capaz de recibir datos, de esta manera ya se tendrá definido uno de los parámetros de empleo de nuestro contador de consumo eléctrico "inteligente".

Para ello, como en el caso anterior, habrá que iniciar la librería VirtualWire [13].Después en el setup habrá que iniciar el receptor. Esta inicialización es similar a la del emisor, con la excepción de que se ha de agregar una última línea en la que indicas que la recepción se inicia.

```
vw_set_ptt_inverted(true);
vw_set_rx_pin(9);
vw_setup(2000);
vw_rx_start();
```

Código 5

Tras la inicialización, se prosigue con la programación en el loop, en este caso lo primero que se ha de hacer es dejar en blanco una variable de carácter para que cuando llegue el dato a recibir se guarde en esta.

A continuación se procede a calcular la longitud del mensaje que se recibe y verificar si hay un dato nuevo disponible que se ha de leer. En caso de que haya un dato nuevo, se procede a su lectura, debido a que se recibe bits, habrá que convertir estos a una variable de carácter, ya que directamente guardarlo en una variable numérica no se puede realizar debido que se ha enviado como carácter.

```
dato_recivido="";
uint8_t buf[VW_MAX_MESSAGE_LEN];
uint8_t buflen = VW_MAX_MESSAGE_LEN;
if (vw_get_message(buf, &buflen))  {
int i;
for (i = 0; i < buflen; i++)
{
dato_recivido+=(char)buf[i];
}
```

Código 6

Se deberá tener el dato recibido guardado como una variable numérica, por lo que habrá que convertir nuestro dato guardado en una variable del tipo carácter a una variable de tipo numérica, esta variable es indiferente que sea del tipo integer o float, esta conversión se hace en la última línea del *Código 6*.

Como se ha comentado previamente, las entradas analógicas de un Arduino poseen un convertidor analógico digital de 8 bits y el cual tiene una tensión de entrada máxima de 5 V, por otro lado tenemos una relación entre la tensión de entrada y la medida, la cual es:

$$I_{medida} = \frac{V_{medida} - {V_{max}}/{2}}{G} * R = \frac{V_{medida} - 2.55}{1.7} * \frac{1000}{33.35} \qquad (3)$$

Se puede observar en la *fórmula 3* la $V_{max}/2$ no corresponde a 2,5 V exactamente. Esto es debido a que se tiene que ajustar para que cuando la sonda mida cero la I_{medida} sea cero, por lo que en este caso tiene como resultado que $V_{max}/2$ tiene que ser 2.55. Por otro lado, la ganancia (G) corresponde a 1.7, y la relación de transformación es de 1000 A por cada 33.35 V, ya que se ha calibrado la sonda. [14]

Como se ve en la *fórmula 3*, se ha de aplicar estas relaciones para la obtención del valor de la tensión medida. Además se ha de tener en cuenta que el valor obtenido es el máximo, por lo que se ha de aplicar la *fórmula 1* para obtener el valor eficaz. Esta operación se realiza en la primera línea de la parte del *Código 7*.

```
dato_enviado=((dato_recivido.toInt()*5./1023)-
2.55)/1.7*1000/33.35/sqrt(2);
sum_dato=dato_enviado+sum_dato;
contador2++;
```

Código 7

La siguiente parte del *Código 7* consiste en sumar todos los datos recibidos, esto se realiza para poder sacar posteriormente la media de los datos recibidos, cada diez minutos, y subir ese dato a la Nube. Lo que nos lleva a la última línea del *Código 7*, la cual es el incremento en uno del contador, cada vez que se ejecute.

Tras la implementación de este código se realiza una prueba para comprobar su correcto funcionamiento, y comprobar hasta que distancia es capaz de realizar la transmisión de datos por radiofrecuencia.

De esta prueba se saca que es capaz de transmitir datos hasta una longitud de 25 metros, lo que se considera una distancia más que razonable. Para la realización de esta prueba se sitúa el emisor en mi puesto de trabajo del laboratorio y se movió el receptor a lo largo del laboratorio y del edificio en el que se sitúa el laboratorio, posteriormente se midió las distancias obtenidas.

Pero esta longitud máxima se reduce debido a que si hay presentes puertas metálicas, al apantallarse la señal en estas. Y al no encontrarse en su envase en el momento en el que se han realizado las pruebas, se supondrá que disminuirá esta longitud.

5.4.2 Conexión a internet

El siguiente proceso a realizar en la programación del Arduino es la conexión a internet. Antes de empezar a realizar el código se ha de elegir el método con el que se accederá a internet, ya que se tienen dos posibilidades:

- Método GET: método generalista para acceder a una URL determinada. Con la desventaja de que los datos se envían desde la URL, por lo que la seguridad es mínima.
- Método POST: método limitado a formularios, con mayor seguridad que el método anterior, ya que el dato a subir no sale en la URL.

Se ha decido emplear el método GET, por su sencillez y debido el método POST puede crear problemas a la hora de subir el dato a Google Drive, como se verá posteriormente.

Tras la elección del método se procede a la programación del código, en esta ocasión se emplean dos librerías, ambas incluidas en la IDE de Arduino. La primera que se emplea es SPI, que establece la comunicación entre el Arduino Uno y la Arduino Ethernet Shield. La segunda librería que se emplea es la de Ethernet, la cual define ciertas funciones, la cual facilita la programación para la conexión a internet.

En el void setup se ha de iniciar la conexión a internet, para ello se intentará iniciar de manera sencilla, es decir únicamente empleando la mac de la Arduino Ethernet Shield. Depende de la configuración de la red por la que te conectas, por ejemplo el Ethernet de la universidad, se deberá emplear la inicialización más compleja como se ve en la segunda línea del *Código 8*.

```
if (Ethernet.begin(mac) == 0) {
Ethernet.begin(mac, ip ,dnserv, gateway , subnet);
}
```

Código 8

En vez de realizar directamente la programación del ingreso a la URL a la que se quiere acceder en el void loop, por claridad se realiza una función externa. Por lo que en el void loop únicamente se tendrá que llamar a esta función, la cual se ha llamado **subir_dato()**.

En esta función, que se muestra en el *Código 9*, lo primero que se hace es asegurarse que se haya desconectado de la anterior URL a la que uno se ha conectado. A continuación, en las siguientes líneas del *Código 9*, se realiza el método GET [15] accediendo a la URL nueva a la que se quiere acceder.

```
void subir_dato(){
client.stop();
if (client.connect(server, 80)) {
client.print("GET /pushingbox?devid=vF0BBAFE762D94B0&status=");
client.print(dato_a_enviar);
client.println(" HTTP/1.1");
client.println("Host: api.pushingbox.com");
client.println("Connection: close");
client.println();
}
}
```

Código 9

5.4.3 Interrupción

El último paso a realizar en la programación de Arduino uno es realizar una interrupción para asegurar la subida de datos cada 10 minutos. Una interrupción se realiza cuando un Timer (contador) ha alcanzado un valor determinado por lo que se desborda

el Timer y esto queda reflejado en un bit, que es el que manda realizar la interrupción. Este contador se ejecutará de manera paralela al resto del programa.

El Arduino Uno tiene 3 Timers. Para ello se empleara el Timer 2 del Arduino Uno, ya que si se reconfigura el Timer 0 no funcionarán los delays ni millis, ni las funciones que empleen estas funciones, y si se emplea el Timer 1 no funcionará la radiofrecuencia.

Una vez que se define el Timer que se va a emplear, se procede a su programación. Para ello antes de nada se ha de definir el compare match register, el valor que te marca cuando se desborda el Timer, con el que se quiere trabajar. Para ello se emplea la *fórmula 4*. Como se busca la mayor frecuencia posible se decide emplear una frecuencia de 1 kHz y el preescaler máximo que se puede dar, como se ve en la tabla 3, será de 64 en la *fórmula 4*.

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk_{T2S}/(No prescaling)
0	1	0	clk_{T2S}/8 (From prescaler)
0	1	1	clk_{T2S}/32 (From prescaler)
1	0	0	clk_{T2S}/64 (From prescaler)

Tabla 3: Preescaler Timer 2 [16]

$$CMR = \frac{velocidad\ de\ reloj\ del\ Arduino}{prescaler * frecuencia\ de\ trabajo} - 1 = \frac{16MHz}{64 * 1kHz} - 1 = 249\ [17] \quad (\mathbf{4})$$

Tras el cálculo del CMR se realiza el código de la interrupción. Para ello en el setup se debe realizar el *Código 10*. Parar la interrupción, por si está ejecutándose por defecto, resetear a 0 los registros e inicializar el contador a 0, se realizan en las 4 primeras líneas del *Código 10*. Posteriormente se define el valor del CMR, se activa el modo CTC, se define el prescaler, se habilita el comparador del Timer y se inicia la interrupción [16].

```
noInterrupts();
TCCR2A = 0;
TCCR2B = 0;
TCNT2  = 0;
OCR2A = 249;
TCCR2A |= (1 << WGM21);
TCCR2B |= (1 << CS22);
TIMSK2 |= (1 << OCIE2A);
interrupts();
```

Código 10

Posteriormente como si se tratase de definir una función se define la acción a realizar cuando se activa la interrupción. En ella se realizará un contador para alcanzar

los 10 minutos, se tiene que calibrar, y cuando se alcance esos 10 minutos se pone a uno una variable que iniciará la subida del dato a internet. Esta función es el *Código 11*.

```
ISR(TIMER2_COMPA_vect)
{
TCNT2=0;
contador1++;
if(contador1==599400)
{
contador1=0;
internet=1;
}
}
```

Código 11

Tras terminar la programación completa se procede ha de realizar una prueba, esta se analizará posteriormente. La programación completa y la definición de sus variables se explican en el Anexo 2.

5.5 Introducción del dato en Google drive

En este apartado se explicará cómo se accede a la hoja de cálculo desde Arduino. El método que se va a emplear es a partir de un formulario de google drive, el cual guarda los datos en una hoja de cálculo [18].

Antes de nada, se ha de crear una cuenta en google, para poder acceder a google drive. Una vez accedido a google drive se deben seguir los siguientes pasos:

1. Crear un formulario en Google drive, para ello se ha de seguir las indicaciones que se observa en la figura 22

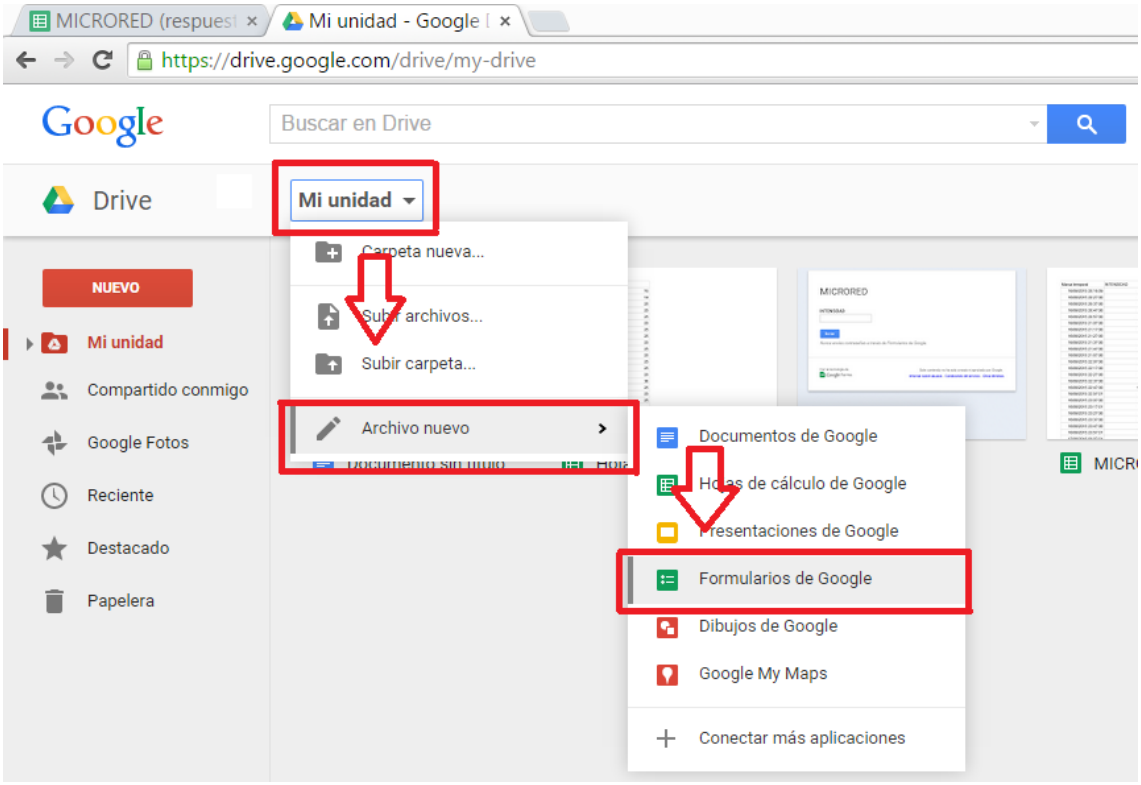


Figura 22: Paso 1 crear formulario

2. Configurar el formulario, para ello se ha de seguir las indicaciones de la figura 23.

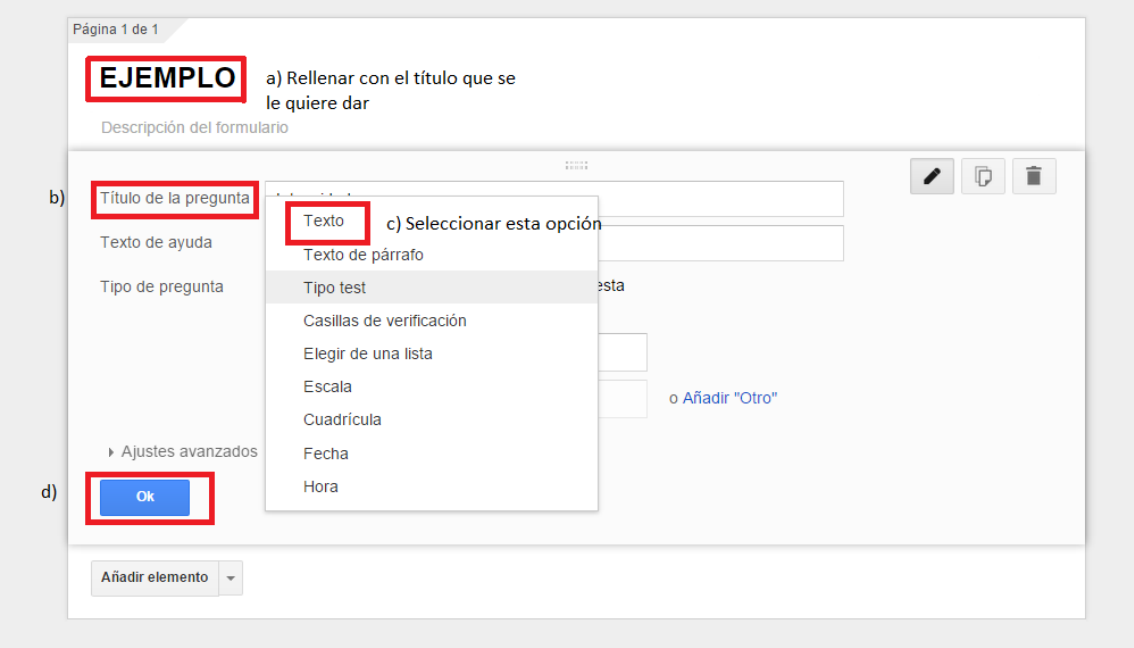


Figura 23: Paso 2 configurar formulario

3. Una vez configurado el formulario, se obtendrá el enlace del formulario en el paso a) de la figura 24, y posteriormente para acceder a la hoja de cálculo se realizara el paso b) de la figura 24.

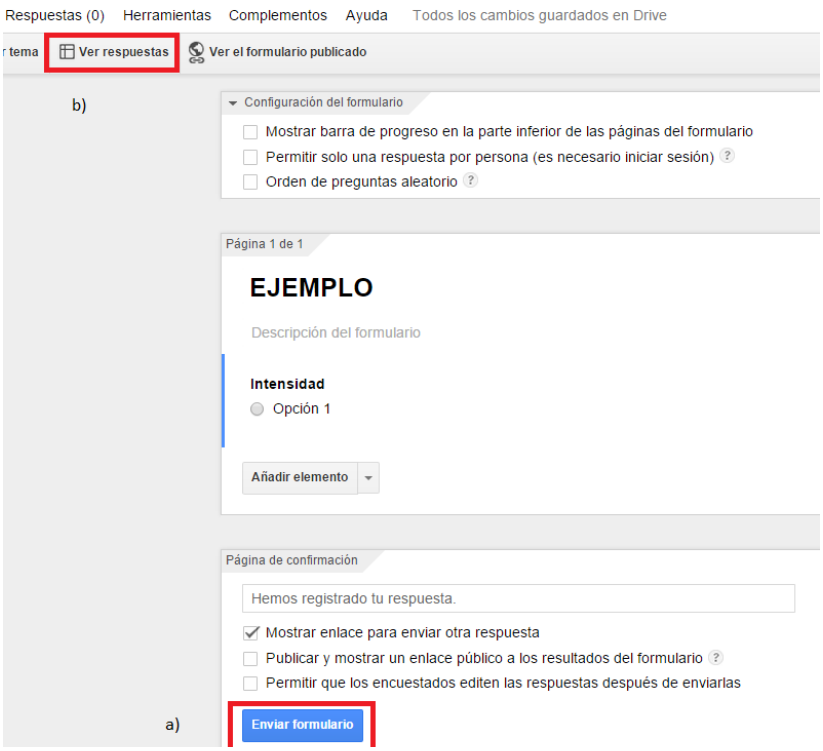


Figura 24: Paso 3 acceder del formulario

4. El siguiente paso a realizar es obtener cierta información del código fuente del formulario que se empleará para crear una URL, la cual al acceder a esta te introduce automáticamente el dato.

Figura 25: Paso 4 obtener información del formulario

Los siguientes pasos se empleará la plataforma pushingbox.com, la cual permite obtener una URL con un servidor que emplee el protocolo HTTP. Esto es necesario debido a que Google Drive emplea el protocolo HTTPS y Arduino no es capaz de trabajar con este protocolo, a causa de que carece de potencia para emplear la librería SSL.

El proceso que realiza pushingbox.com internamente es mediante su servidor realizar el trabajo que Arduino no es capaz de realizar, es decir tu accedes a su servidor y su plataforma al recibir el dato lo envía al formulario de Google Drive.

Se podría realizar esto mismo programando en php un código que realizase eso, por ejemplo creando un blog y programándolo. Pero debido a que se requieren conocimientos informáticos avanzados se descarta esta opción.

5. A continuación se ha de configurar el servicio que se quiere emplear en pushingbox.com. para ello se ha de realizar los pasos que se ven en la figura 26 accediendo a la pestaña MyService.

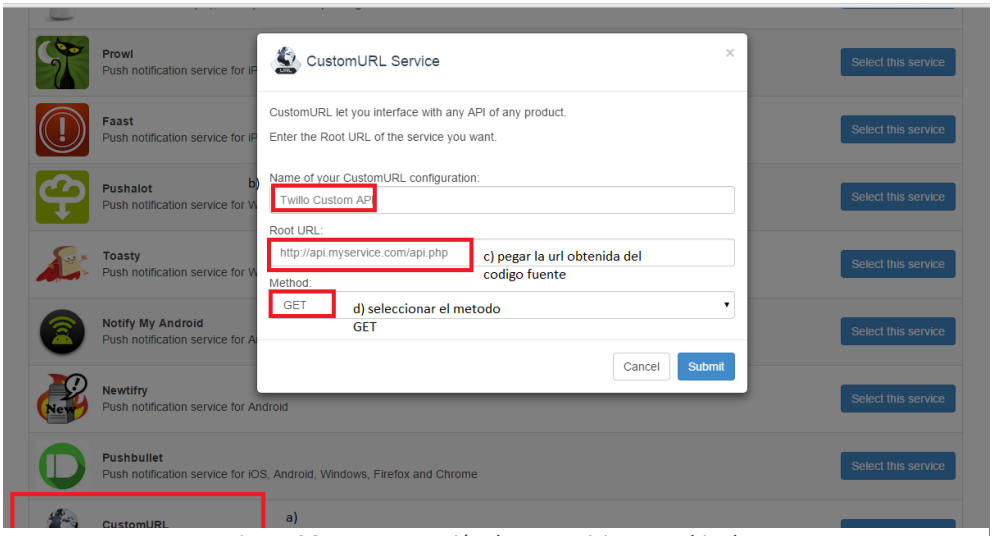


Figura 26: Paso 5 creación de un servicio en pushingbox

6. Tras configurar el servicio, se deberá definir el escenario, para ello se debe acceder a la pestaña de servicios, y darle un nombre al servicio nuevo, al darle a *añadir*, te redirecciona a una nueva sección en la cual se ha de seleccionar *añadir acción.* Al realizar esto se accede a una nueva sección donde se termina de configurar el escenario, como se observa en la figura 27.

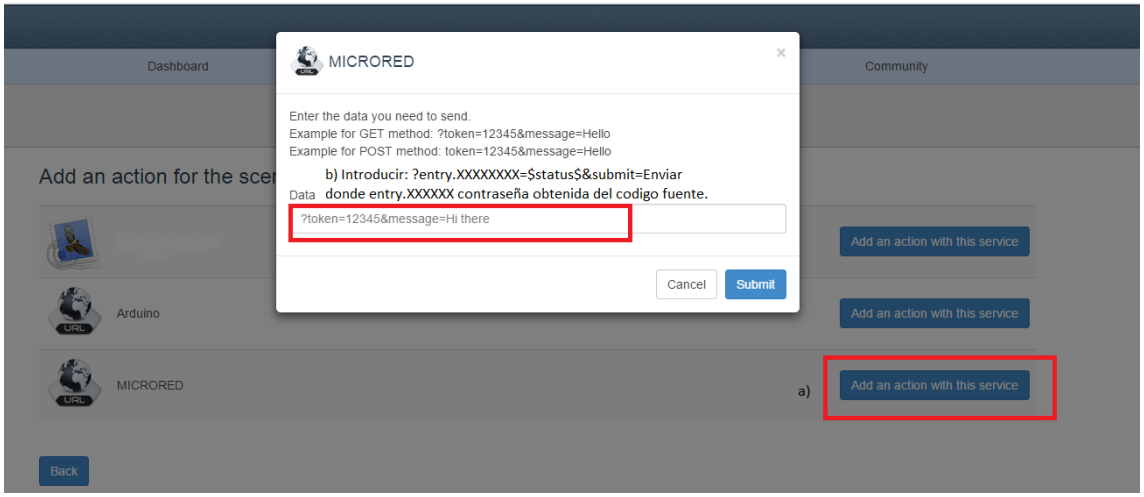


Figura 27: Paso 6 definir un escenario

7. Por último para la obtención de la URL, se ha de volver a la pestaña de escenarios para obtener la contraseña, como se observa en la figura 28.

Figura 28: Paso 7 obtención contraseña

La dirección URL, que se ha de emplear quedará con la siguiente forma: http://api.pushingbox.com/pushingbox?devid=XXXXXXXXXXXXXX&status="dato_a_enviar", donde en "dato_a_enviar" se ha de poner el dato que se quiere subir, y se ha de sustituir las x por la contraseña obtenida en el paso 7.

6 ENSAYO FINAL

Tras la finalización de la programación de ambos Arduinos se decide realizar un ensayo, midiendo la intensidad de salida del inversor de la microrred de la UPNA. Para ello se coloca el sensor en la segunda fase de la salida del inversor. Se ha estado sensado durante 6 días, pero se analizará los resultados de medio día, ya que como se comentará posteriormente han ocurrido diversos fallos a lo largo del tiempo.

Lo primero que cabe destacar es que se tiene un retraso acumulativo de 42 milisegundos, como se ve en la primera columna de la tabla 4. Esto es debido a que al saltar la interrupción no se inicia la subida del dato a Google Drive, sino que se termina de ejecutar el loop, y cuando termine de ejecutarse, en el nuevo inicio del loop se enviará el dato y además la propia interrupción tiene un retraso.

Ensayo Intensidad de salida del inversor de la microrred			
dd/mmm/aa hh:mm:ss	Imedida_arduino (A)	Imedida_micro-rred (A)	Error (%)
17/06/2015 0:17	7,48	7,350185	1,766146022
17/06/2015 0:27	7,49	7,369806667	1,630888553
17/06/2015 0:37	7,48	7,34629	1,820102392
17/06/2015 0:47	7,48	7,356841667	1,674065297
17/06/2015 0:57	7,44	7,34017	1,360050244
17/06/2015 1:07	7,46	7,348905	1,511721814
17/06/2015 1:17	7,46	7,343455	1,587059497
17/06/2015 1:27	7,48	7,337572621	1,941069429
17/06/2015 1:37	7,52	7,355295	2,239271165
17/06/2015 1:47	7,52	7,360728333	2,163803084
17/06/2015 1:57	7,52	7,35337	2,266035845
17/06/2015 2:07	7,5	7,339361667	2,188723497
17/06/2015 2:17	7,55	7,381055092	2,288899162
17/06/2015 2:27	7,5	7,353837675	1,987565284
17/06/2015 2:37	7,48	7,34725	1,806798462
17/06/2015 2:47	7,45	7,354771667	1,294783001
17/06/2015 2:57	7,48	7,37489	1,425241597
17/06/2015 3:07	7,49	7,381161667	1,474542061
17/06/2015 3:17	7,45	7,363021703	1,181285356
17/06/2015 3:27	7,46	7,355094074	1,42630298
17/06/2015 3:37	7,47	7,355208333	1,560685455
17/06/2015 3:47	7,51	7,39119	1,60745428
17/06/2015 3:57	7,53	7,410295	1,615387781
17/06/2015 4:07	7,5	7,36721797	1,802336112
17/06/2015 4:17	7,49	7,365333333	1,692614048

17/06/2015 4:27	7,78	7,607445	2,268238548
17/06/2015 4:37	8,95	8,794935	1,763117067
17/06/2015 4:47	8,89	8,803875	0,978262413
17/06/2015 4:57	8,93	8,818196995	1,267866947
17/06/2015 5:07	8,21	8,08571	1,537156292
17/06/2015 5:17	7,71	7,633551667	1,001477905
17/06/2015 5:27	7,65	7,493561667	2,087636564
17/06/2015 5:37	8,02	7,896209732	1,567717584
17/06/2015 5:47	7,63	7,508240803	1,621674111
17/06/2015 5:57	7,6	7,430838063	2,276485305
17/06/2015 6:07	7,5	7,401895175	1,325401441
17/06/2015 6:17	7,45	7,360498333	1,215972922
17/06/2015 6:27	7,63	7,525735	1,385446073
17/06/2015 6:37	7,51	7,388228715	1,648179695
17/06/2015 6:47	8,85	8,790233333	0,679921276
17/06/2015 6:57	9,23	9,126111667	1,1383636
17/06/2015 7:07	10,01	9,915716667	0,950847392
17/06/2015 7:17	8,97	8,845160267	1,411390287
17/06/2015 7:27	9,15	9,02416	1,394478821
17/06/2015 7:37	9,17	9,038398333	1,456028622
17/06/2015 7:47	9,87	9,812133333	0,589746029
17/06/2015 7:57	10,45	10,40439333	0,43834047
17/06/2015 8:07	10,05	10,00983167	0,4012888
17/06/2015 8:17	11,34	11,33366389	0,055905191
17/06/2015 8:27	10,94	10,91901169	0,192218073
17/06/2015 8:37	10,43	10,460055	0,287331185
17/06/2015 8:47	11,32	11,30667333	0,117865497
17/06/2015 8:57	9,36	9,372026667	0,128325143
17/06/2015 9:07	8,38	8,561171667	2,116201774
17/06/2015 9:17	8,87	8,992333333	1,360418134
17/06/2015 9:27	7,73	7,67563606	0,708266253
17/06/2015 9:38	11,68	11,68205167	0,017562554
17/06/2015 9:47	10,56	10,58128	0,201109885
17/06/2015 9:57	11,66	11,71655927	0,482729308
17/06/2015 10:07	11,67	11,70409	0,291265703
17/06/2015 10:17	11,66	11,70789983	0,409124042
17/06/2015 10:27	11,66	11,68853333	0,244113889
17/06/2015 10:37	11,67	11,66111167	0,076222007
17/06/2015 10:47	11,76	11,695723	0,549576969
17/06/2015 10:57	11,74	11,726055	0,118923201

17/06/2015 11:07	11,72	11,73268	0,1080742
17/06/2015 11:17	11,72	11,73993667	0,169819201
17/06/2015 11:27	11,74	11,74289	0,024610637
17/06/2015 11:37	11,71	11,65230167	0,495166835
17/06/2015 11:47	11,69	11,643015	0,403546676
17/06/2015 11:57	11,55	11,58285667	0,283666349
17/06/2015 12:07	11,56	11,57051667	0,090891937

Tabla 4: Resultado Ensayo microrred

Por otro lado también hay un retraso o adelanto no acumulativo causado por la velocidad de subida, por lo que se producen variaciones de un segundo que posteriormente se corrigen.

Por último se puede observar el error en la medición es de en torno a 1%, pero conforme más se aleje la medición de 15 A se producirá un error, debido a que se ha tomado la relación de transformación como una media. Se considera que este error de precisión es bueno, ya que como se observa en la figura 29 la diferencia es mínima.

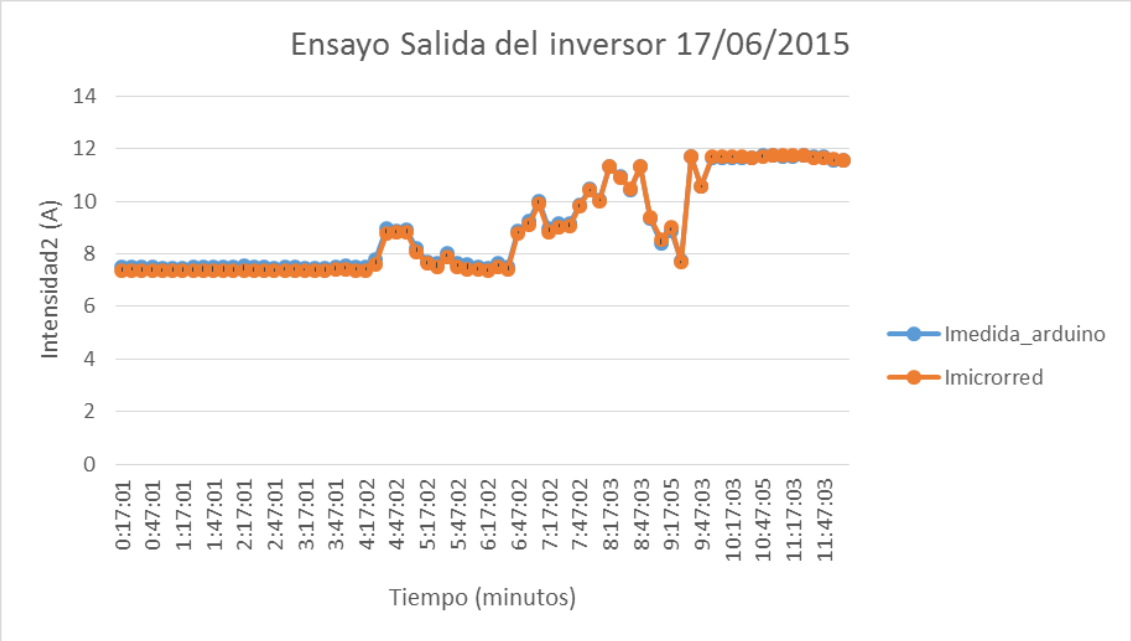


Figura 29: Ensayo microrred

Durante el ensayo se observó que se producen errores en la captación de datos, producidas por el mal contacto que hace las patillas del módulo receptor en el Arduino Ethernet Shield. Además se vio que al perder la conexión a internet el Arduino Uno se reiniciaba.

7 COMPARATIVA ENTRE ARDUINO Y RASPBERRY PI

A continuación, se realiza una comparativa el procesamiento de la señal mediante arduino y el procesamiento mediante Raspberry Pi [19]. Para ello se comparan diferentes características que aportan el empleo de cada alternativa:

- El empleo de Arduino emplea dos módulos diferentes, uno para el sensado y otro para la subida del dato a la hoja de cálculo almacenada en Google-Drive. Estos módulos se comunican mediante radiofrecuencia. Por el contrario, la Raspberry Pi únicamente necesita un módulo, el cual realiza ambas operaciones. Esto conlleva a que el empleo de Arduino requiera mayor espacio, aunque éste esté distribuido en diferentes zonas.
- En cuanto a la conexión a internet, se realiza por medio de WiFi por parte de la Raspberry Pi, y mediante conexión de cable Ethernet por parte de Arduino. En el caso de la Raspberry Pi, la distancia a la que se podrá colocar el contador del router dependerá del alcance del WiFi. Con Arduino, el módulo que habrá que conectar a internet se podrá colocar cerca del router, por lo que el cable Ethernet necesario será corta longitud. Pero la distancia a la que se podrá colocar un módulo del otro, y por lo tanto, la distancia del contador con el router quedará limitada por la capacidad de la radiofrecuencia.
- A pesar de que el Arduino utiliza dos módulos que se deben alimentar por separado, el consumo de la Raspberry Pi es mayor que el de los dos módulos de Arduino juntos. Esto es debido a que el adaptador WiFi utilizado en la Raspberry Pi para conectarnos a internet obliga a la utilización de un cargador de un mínimo de 1 A (en el trabajo se ha usado un cargador de 2 A) para su correcto funcionamiento. Por otro lado, el módulo de sensado de Arduino consume 40 mA y se emplea un cargador con capacidad de proporcionar 1.3 A; el módulo de envío de datos consume una media de 220 mA, y se empleará un cargador similar al otro módulo.
- Debido a que el convertidor analógico-digital utilizado con Arduino procesa peor la señal comparado con el convertidor analógico-digital empleado con la Raspberry Pi, en la Raspberry Pi se obtiene una mejor precisión en los datos. También se ha contemplado la opción de que la diferencia de error es debida a la forma de calcular el valor eficaz de la corriente. Arduino emplea la fórmula uno y la Raspberry Pi emplea la fórmula dos.
- A la hora de subir datos, Arduino tiene un error acumulativo de 42 ms, lo que se traduce en 6 segundos al día. Por el contrario, la Raspberry Pi cumple el objetivo de subir los datos cada 10 minutos, pero en ocasiones se produce un error que para el sistema.

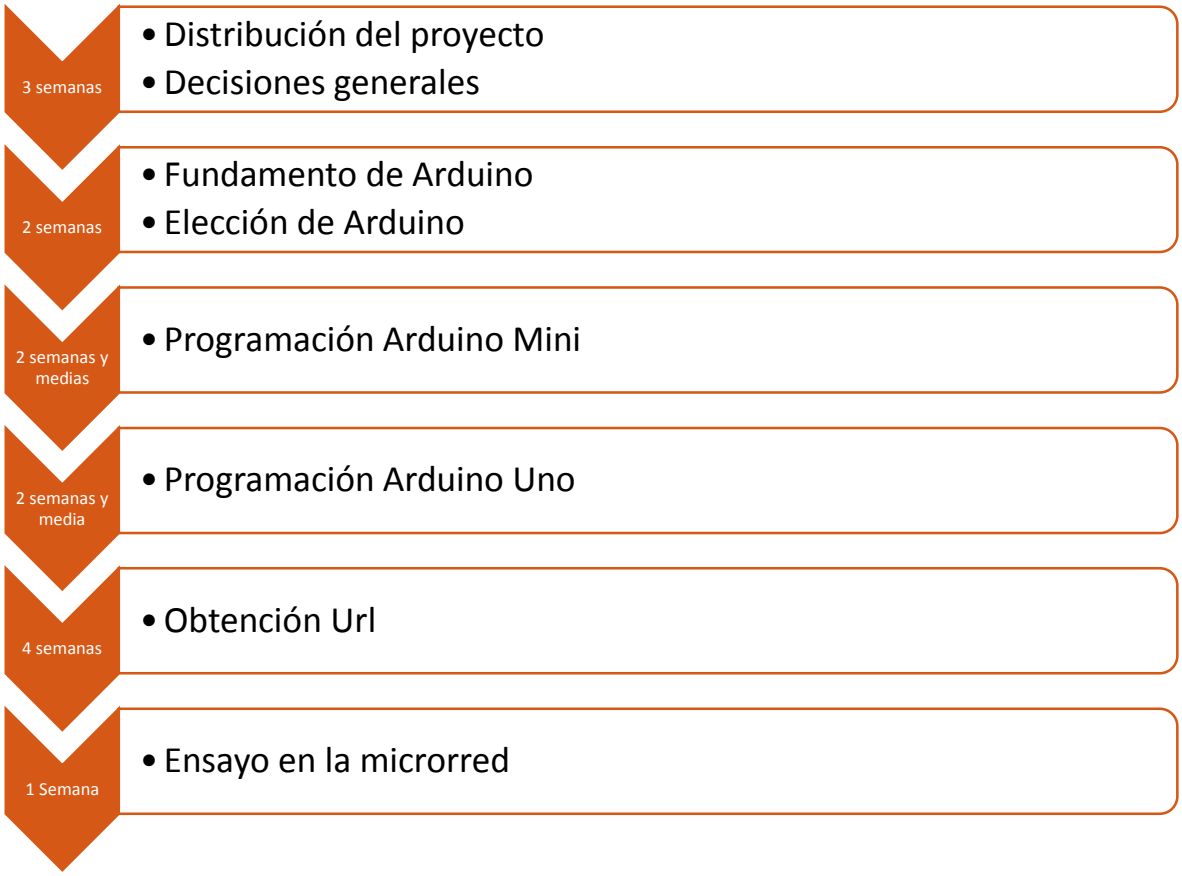
- Por último, desde un punto de vista económico, el empleo de Raspberry Pi resulta más económico que el de Arduino, aunque la diferencia de precio no es muy alta, ya que sus precios son 91.46 € y 109.18 € respectivamente. [20]

Para concluir, se emplea la tabla 5 para poder analizar de forma rápida las ventajas y desventajas de cada sistema de procesado de datos:

	Arduino	**Raspberry Pi**
Tamaño	Mayor	Menor
Tipo conexión	Ethernet	WiFi
Alimentación	<500 mA	>1 A
Precisión	Menor	Mayor
Tiempo de subida	No exacto	Se interrumpe
Precio	109,18 €	91,46 €

Tabla 5: Comparación entre Arduino y Raspberry Pi

8 LINEA TEMPORAL



9 CONCLUSIONES

Como conclusión final, una vez ensamblado todo y ensayado, es que se ha conseguido desarrollar de dos formas diferentes un contador de consumo eléctrico inteligente, que cumple con los objetivos impuestos.

En la parte desarrollada en esta memoria se ha conseguido:

- Precisión de un 1 % de media a la hora de obtener el valor eficaz.
- Tener un dispositivo que procese los datos de internet y los suba a la Nube.
- Subir el dato de forma aproximadamente periódica de 10 minutos.
- Optimizar el coste del producto.

10 LÍNEAS FUTURAS

Las mejoras que se podrían introducir en este proyecto son:

- Mejorar la conexión de las patillas entre el módulo receptor y la Arduino Ethernet Shield.
- Lograr la subida exacta en 10 minutos de los datos.
- Obtener mayor precisión tomando la relación de transformación, no como una constante, sino como una función.

BLIBLIOGRAFÍA

[1] CONTADORELECTRICIDAD http://www.contadorelectricidad.com/owl-micro.html

[2] CLIENSOL http://www.cliensol.es/envir.html

[3] EFERGY http://efergy.com/es/engagekit-hub

[4] ENGAGE.EFERGY https://engage.efergy.com/dashboard

[5] ELVERDADEROCOLORDELDINERO

http://elverdaderocolordeldinero.blogspot.com.es/2012/10/medidores-de-consumo-electrico-para-el.html

[6] REVUELTA IRRISARI, JOSEBA Monitorización del consumo eléctrico de un hogar: Sensado y acondicionamiento de la corriente 2015, Upna

[7] LEGARREA OYARZUN, ARITZ Monitorización del consumo eléctrico de un hogar: Procesado de datos mediante Raspberry Pi 2015, Upna

[8] BARRIOLA HERNANDORENA, NAIA Monitorización del consumo eléctrico de un hogar: Visualización vía web 2015, Upna

[9] dropbox.com

[10] GOOGLE https://www.google.com/intl/es_es/drive/using-drive/

[11] NEWS.MICROSOFT http://news.microsoft.com/presskits/onedrive/

[12] arduino.cc

[13] PANAMAHITEK

http://panamahitek.com/transmitiendo-recibiendo-datos-por-ondas-de-radio-con-arduino/

[14] REVUELTA IRRISARI, JOSEBA Monitorización del consumo eléctrico de un hogar: Sensado y acondicionamiento de la corriente 2015, Upna

[15] Arduino.cc

[16] INSTRUCTABLES

http://www.instructables.com/id/Arduino-Timer-Interrupts/step1/Prescalers-and-the-Compare-Match-Register/

[17] ATMEL ATmega48A/PA/88A/PA/168A/PA/328/P 8-BIT MICROCONTROLLER WITH 4/8/16/32KBYTES IN-SYSTEM PROGRAMMABLE FLASH DATASHEET

[18] INSTRUCTABLES

http://www.instructables.com/id/Post-to-Google-Docs-with-Arduino/?ALLSTEPS

[19] LEGARREA OYARZUN, ARITZ Monitorización del consumo eléctrico de un hogar: Procesado de datos mediante Raspberry Pi 2015, Upna

[20] REVUELTA IRRISARI, JOSEBA Monitorización del consumo eléctrico de un hogar: Sensado y acondicionamiento de la corriente 2015, Upna

ANEXO 1

```
/*-----------------------------------------------------
------------------Arduino Pro Mini-------------------
-------------Proyecto final de carrera:--------------
--Monitorización del consumo eléctrico de un hogar:--
---------Procesado de datos mediante Arduino---------
-----------------------------------------------------
----------------Versión:2.1--------------------------
-------------Fecha:20/06/2015-----------------------
-----------Víctor Erice Carbonero--------------------
-----------------------------------------------------*/

// LIBRERIA INCORPORADA
#include <VirtualWire.h>        // Librería obtenida de "pagina web"
para la transmisión de datos por radiofrecuencia


//CONSTANTES DEL PROGRAMA

#define N_CICLOS 75  // numero de ciclos que queremos tomar para hacer
la media de los máximos de tensión

// VARIABLES GLOBALES DEL PROGRAMA

int analog_pin = A3;            // Pin de entrada analogica por la
cual entra el señal del sensor (bits)
int tension;                    // Valor de la entrada analógica, esta
en bits pero representa la tensión del sensor
int tension_max_ciclo;          // Valor máximo del ciclo, bits repre-
sentando tensión
float sum_tension;                // Suma de los máximos de XXXXXXXXXX
ciclos, bits representando tensión
int media_tension;              // Media de los máximos de
xxxxxxxxxxxxx ciclos, bits representando tensión

int i;
int j;

String str;
char b[5];                      // Tensión media de los máximos con-
vertida a una variable de tipo caracter para su envio por radiofre-
cuencia



void setup()
{
  Serial.begin(9600);         // Definir velocidad de transmision de
datos desde el puerto USB
  vw_set_ptt_inverted(true);  // Requerido por el modulo RF
  vw_setup(2000);            // Velocidad de conexion bps
  vw_set_tx_pin(3);        // Pin de salida digital en el que conecta-
mos la patilla data del transmisor
}


// Función para la adquicisión del maximo por ciclo
void sensado(){
```

```
  tension_max_ciclo=0; // Inicializar a cero la tensión maxima del ci-
clo

  for (i=1;i<21;i++){ //
    tension = analogRead(analog_pin);
    if (tension>tension_max_ciclo){
      tension_max_ciclo=tension;
    }
    delayMicroseconds(883);
  }
}


void loop()
{
 sum_tension=0;
 for (j=0;j<N_CICLOS;j++){
   sensado();
   sum_tension=sum_tension+tension_max_ciclo;
 }
 media_tension=sum_tension/N_CICLOS;
 str=String(media_tension);
 str.toCharArray(b,5);

 vw_send((uint8_t *)b, strlen(b));
 vw_wait_tx();       // Esperamos a que termine de enviar el mensaje

}
```

ANEXO 2

```
/*-----------------------------------------------------
--------------------Arduino Uno----------------------
-------------Proyecto final de carrera:--------------
--Monitorización del consumo eléctrico de un hogar:--
---------Procesado de datos mediante Arduino---------
-----------------------------------------------------
----------------Versión:3.1--------------------------
-------------Fecha:20/06/2015------------------------
-----------Víctor Erice Carbonero--------------------
-----------------------------------------------------*/

//LIBRERIA INCORPORADA

#include <SPI.h>
#include <Ethernet.h>
#include <VirtualWire.h>


//CONSTANTES DEL PROGRAMA

#define V_ESCALA 5 // Fondo de escala de la tensión (V)
#define BITS_ESCALA 1023 // Fondo de escala del convertidor A/D (bits)
#define V_CERO 2.55 // Tension que da la sonda cuando hay 0A (V)
#define GANANCIA 1.7 // Ganancia del circuito de acondicionamiento
#define RELACION_TRANSFORMACION 1000/33.35 // Relación de transforma-
ción de la sonda (A/V)
#define CMR 249 // Valor que marca cuando se desborda el Timer 2
#define TIEMPO_SUBIDA 599400 // Cada cuantos milisigundos tiene que
subirse un dato a internet (ms)

// VARIABLES GLOBALES DEL PROGRAMA

byte mac[] = {0x90,0xA2,0xDA,0x0F,0x72,0xBD};  //MAC del Arduino
Ethernet Shield
byte ip[] = {172,18,93,26};  //IP
byte subnet[] = {255,255,254,0}; // Máscara de la subred
byte gateway[] = { 172,18,64,254}; // Puerta de enlace
byte dnserv[]={130,206,159,1}; // DNS
char server[] = "api.pushingbox.com";    // Dirección del servidor al
que se conecta



float contador_internet=0; // Contador para acctivar el envio de datos
a internet
int contador_media=0; // Contador para poder sacar la media de los va-
lores máximos
int internet=0;

float dato_guardado; // Dato que se guarda al recivirlo
float sum_dato=0; // Sumatorio de datos guardados
float dato_a_enviar; // Dato que se debe enviar
String dato_recivido; // Dato recivido por radiofrecuencia
```

```
EthernetClient client; // definir la variable cliente

void setup() {
  Serial.begin(9600);
  // Iniciar receptor de radiofrecuencia
  vw_set_ptt_inverted(true);
  vw_set_rx_pin(9); //Pin donde se conecta el Arduino
  vw_setup(2000); //Tiempo de espera
  vw_rx_start(); //Se inicia la recepción de datos
  // Iniciar conexión a internet
  if (Ethernet.begin(mac) == 0) {
    Ethernet.begin(mac, ip ,dnserv, gateway , subnet);
    }
  // esperar un segundo para la conexión correcta a internet:
  delay(1000);

  // Configurar Timer 2 para la interrupcion
  noInterrupts();
  //set timer2 interrupt at 8kHz
  TCCR2A = 0;// set entire TCCR2A register to 0
  TCCR2B = 0;// same for TCCR2B
  TCNT2  = 0;//initialize counter value to 0
  // set compare match register for 8khz increments
  OCR2A = CMR;// = (16*10^6) / (8000*8) - 1 (must be <256)
  // turn on CTC mode
  TCCR2A |= (1 << WGM21);
  // Set CS21 bit for 8 prescaler
  TCCR2B |= (1 << CS22);
  // enable timer compare interrupt
  TIMSK2 |= (1 << OCIE2A);
  interrupts();
}

// Función de la interrupción
ISR(TIMER2_COMPA_vect)
{
  TCNT2=0;
  contador_internet++;
  if(contador_internet==TIEMPO_SUBIDA)
    {
      contador_internet=0;
      internet=1;
     }
}

void loop()
{
  // Envio de dato a la Nube
  if (internet==1)
  {
    dato_a_enviar=sum_dato/contador_media;
    internet=0;
    contador_media=0;
    sum_dato=0;
    Serial.println("subido");
    subir_dato();
    if (!client.connected()) client.stop();

   }
  // Recepción del dato por radiofrecuencia
  dato_recivido="";
```

```
  uint8_t buf[VW_MAX_MESSAGE_LEN]; //Tamaño del mensaje
  uint8_t buflen = VW_MAX_MESSAGE_LEN;
  if (vw_get_message(buf, &buflen)) //Se verifica si hay //mensaje
disponible para ser leido
  {
    int i;
    for (i = 0; i < buflen; i++) //Se leen todos los caracteres
    {
      dato_recivido+=(char)buf[i]; //Se guarda la información en una
//matriz
    }
    //Serial.println("dato");
    dato_guardado=((dato_recivido.toInt()*V_ESCALA/(BITS_ESCALA))-
V_CERO)/(GANANCIA)*RELACION_TRANSFORMACION*1./sqrt(2);
    sum_dato=dato_guardado+sum_dato;
    Serial.println(dato_guardado);
    contador_media++;
  }

}


// Función de envio de dato a la Nube
void subir_dato(){
  client.stop();
  if (client.connect(server, 80)) {
    //Serial.println("connected");
    // Make a HTTP request:
    client.print("GET /pushingbox?devid=vF0BBAFE762D94B0&status=");
    client.print(dato_a_enviar);
    client.println(" HTTP/1.1");
    client.println("Host: api.pushingbox.com");
    client.println("Connection: close");
    client.println();
  }
  }
```