Universidad de Alcalá
Escuela Politécnica Universidad Superior

Trabajo Fin de

Grado

Demostrador de un planificador agregador de la demanda de electricidad en una comunidad de consumidores

Autor: Daniel Fernández Bodas

Tutor: Esther Palomar González

2022

UNIVERSIDAD DE ALCALÁ
Escuela Politécnica Superior

Grado en Ingeniería en Electrónica de comunicaciones

Trabajo Fin de Grado

Diseño e implementación de edificio de electricidad agregada

Autor: Daniel Fernández Bodas

Tutor: Esther Palomar González

TRIBUNAL:

Presidente: Álvaro Hernández Alonso

Vocal 1º: José Manuel Rodríguez Ascariz

Vocal 2º: Esther Palomar González

FECHA: 31-03-2022

Agradecimientos

A mis padres, por apoyarme, por ayudarme y por darme todo lo que soy, y a mi hermana por siempre estar ahí.

A mis amigos de la universidad, en especial a Cris y Dani, por acompañarme durante todos estos años y por tantas cosas vividas juntos.

A mis amigos de Guada, con los que paso las horas, me hacen vivir momentos tan felices y hacen que cada día merezca la pena.

A mi tutora Esther, por permitirme hacer este trabajo y su infinita paciencia conmigo.

Resumen

Este TFG tiene como objetivo el montaje de un demostrador de un sistema que controla la demanda agregada de una comunidad con el fin de demostrar que conjuntamente se puede ser más verde, sostenible y lograr una demanda de energía renovable. Las comunidades sostenibles son cada vez más demandadas en la sociedad debido a la calidad de vida que generan gracias a la reducción del impacto medioambiental y el ahorro de energía.

Este montaje se hará mediante juegos de piezas Lego, en los que se incluirán elementos electrónicos típicos de cada establecimiento (casas, talleres, etc.) y en los que se podrá realizar una planificación horaria de cada elemento.

Las etapas que se han seguido han sido: montaje del demostrador, inclusión de los elementos electrónicos, diseño del software para facilitar la programación horaria, verificación y pruebas funcionales.

El montaje se valida con una serie de escenarios que simulan tipos de comunidad y de demanda.

Palabras clave: Planificación de demanda, electricidad, eficiencia, maqueta, comunidad.

Summary

This final year Project develops a demonstrative mockup of an energy-efficiency system in a community of electricity consumers that aggregate their demand to jointly achieve more sustainable goals. Sustainable communities are emerging fast in society due to the benefits it generates such as the reduction of environmental impact and energy savings.

The mockup is constructed over two Lego sets, in which light-weight cost-effective electronic elements are mounted to simulate household appliances and devices. RPI boards are programmed to light on/off the corresponding devices according to the community schedule. The community schedule is configured as a web-service hosted on a RPI board and trained through a series of scenarios that simulate community and demand types.

Therefore, the stages that this project conducted were: 1) assembly of the mockup demonstrator, 2) inclusion of electronic elements, 3) software design to facilitate time scheduling, and 4) verification and functional testing.

Key words: Demand scheduler, electricity, efficiency, mockup, community.

Acrónimos

RPI (Raspberry Pi)

LED (Light Emitting Diode)

ML (Machine Learning)

SoC (System on a Chip)

CPU (Central Processing Unit)

RAM (Random Access Memory)

SD (Secure Digital)

GPIO (General Purpose Input/Output)

HTML (HyperText Markup Language)

SQL (Structured Query Language)

PHP (Hypertext Preprocessor)

MYSQL (My Structured Query Language)

IP (Internet Protocol)

CSV (Código Seguro de Verificación)

Índice general

1- Introducción

Este TFG consiste en mostrar un sistema que controla la demanda de electricidad de una comunidad de consumidores que cooperan gracias a un agregador. Está incluido en el proyecto "Eneff-Pilot", coordinado por el Departamento de Electrónica de la Universidad de Alcalá y que pretende promover cambios de comportamiento en comunidades con intereses comunes como conseguir objetivos de sostenibilidad, autoconsumo y eficiencia.

1.1- Motivación y objetivos

La motivación principal para escoger este trabajo fue el interés en el mundo de la domótica, es decir, los sistemas capaces de automatizar una vivienda o edificación de cualquier tipo, aportando servicios de gestión energética, seguridad, bienestar y comunicación.

Además, influyó también la afición por realizar proyectos prácticos, y por adquirir e incrementar mis aptitudes y conocimientos de programación.

Este trabajo tiene 3 objetivos:

- **Estudio:** En este trabajo se pretende mostrar la importancia de estas comunidades sostenibles en el futuro, pero también en el presente, especialmente en el desarrollo de programas para el ahorro energético y económico. Para ello, se estudia la creación de un sistema sencillo de control de los dispositivos y electrodomésticos a nivel de consumidor, así como del control conjunto de varios consumidores. Se incluirá tecnología en aparatos domésticos corrientes como puede ser una lavadora o un horno, entre otros, y se establecerá una programación horaria de manera remota para posibilitar el mayor consumo del suministro proveniente de renovables y el autoconsumo. Además, atendiendo a dicha programación y los patrones de demanda, se podrán analizar los distintos comportamientos de comunidades como, por ejemplo, *Demanding*, con mucho volumen y picos de consumo, *Busy* con mucho volumen, pero sin picos a lo largo del día, o *Concerned,* con poco volumen [18]. Esta programación la hará de forma centralizada un agregador con el fin de controlar la demanda completa de la comunidad y poder optimizar la planificación del consumo en el día de antes.

- **Diseño y montaje:** Se diseña y monta una maqueta en bloques de Lego para visualizar el funcionamiento del sistema explicado con anterioridad. Para ello se simularán aparatos domésticos a través de componentes electrónicos conectados a la RPI *board* y en *protoboards* en las que se agrega y planifica la

electricidad asignada a cada componente. Es decir, estos aparatos están directamente conectados a un microcontrolador que será el encargado de apagarlos y encenderlos atendiendo a la programación horaria establecida.

- **Validación:** Se estudian distintos escenarios de consumidores y de comunidades y se comprobará que los aparatos electrónicos se encienden y apagan en el horario establecido. Se hará con la ayuda de un software web al que tendrá acceso un administrador (que simula al agregador central) y a través del cual podrá cargar los escenarios que irán al microcontrolador.

1.2- Metodología

La elaboración de este proyecto se hará en cuatro partes: Montaje de la comunidad, inclusión de los elementos electrónicos, cableado, y creación de un *webserver* para ayudar a establecer la programación horaria.

La primera parte trata de afrontar cómo se construirá la maqueta. Para ello, se decide que la mejor forma es hacerlo mediante piezas Lego debido a la sencillez para su elaboración gracias al guiado paso a paso que ofrecen y, en caso de necesitarse, tener que modificar la estructura inicial para adaptarse a las necesidades que se requieran.

Para la elección del Lego, se busca que incluya varios establecimientos para poder incorporar varios elementos electrónicos. Además, se intentará que incluya varias ventanas para que dichos elementos puedan ser vistos sin problemas y, de esa manera, facilitar el cableado posterior. Tras varias opciones, finalmente se eligen dos juegos de piezas Lego, los mostrados en la Figura 1, los cuales se modificará ligeramente la estructura de fábrica para juntarlos en una sola comunidad.

Figura 1: Legos escogido para la construcción de la comunidad.

Tras la elección del edificio, se escogerán los elementos electrónicos a incluir. Estos elementos simularán aparatos electrónicos típicos que se pueden encontrar en los establecimientos de una comunidad (casas, tiendas, talleres, etc.) como pueden ser lavadoras, calefacciones o ventiladores. Dichos aparatos se harán mediante diodos leds de diferentes colores y motores. Los motores utilizados serán los "28BYJ-48", mostrados en la Figura 2.

Figura 2: Motor "byj48" y su adaptador

El motor "byj48" es un motor muy típico en proyectos pequeños de robótica ya que es muy económico y sencillo de usar. Es un motor de cuatro fases diseñado para girar un paso cada vez que se alimenta una fase. Tiene 3 formas de alimentarse [1]: excitando dos bobinas a la vez, donde tendríamos más velocidad y más par pero más consumo, excitando solo una bobina, con menos consumo y menos par, o con medios pasos, es decir, una forma intermedia de los dos casos anteriores. Para este proyecto, se utilizará este último método, mostrado en la Figura 3.

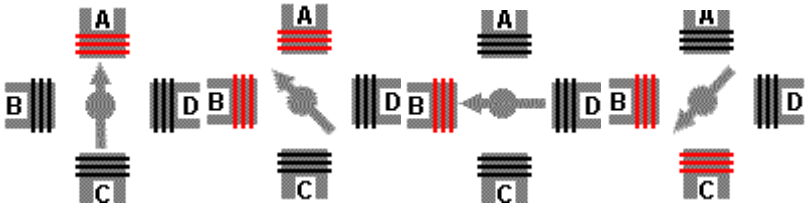


Figura 3. Pasos para la alimentación del motor "byj48"

Una vez que se sabe cuáles serán los elementos electrónicos, se ha de ver cómo se controlan. Para ello, lo primero en lo que se piensa es en un microcontrolador, controlando a través de sus periféricos de entrada/salida los elementos electrónicos, pero dado que posteriormente hay que desarrollar un *webserver*, lo ideal sería hacerlo todo en un mismo dispositivo. Y, sin duda, la mejor forma posible para este desarrollo es la RPI (Raspberry Pi).

Figura 4. RPI 3b.

La RPI es un ordenador de bajo coste y tamaño reducido, como se puede ver en la Figura 4. Dispone de un SoC, CPU, memoria RAM y puertos de entrada/salida y una ranura SD para almacenamiento, donde se instalará el software [2]. Dispone, además, de 26 pines GPIO. Debido a que cada motor necesita 4 pines GPIO y que por cada establecimiento se prevé poner 3 o 4 elementos, se prevé que se necesiten 2 o 3 RPIs.

1.3- Trabajos relacionados

Para la elaboración de este trabajo, se han buscado proyectos relacionados con las comunidades energéticas atendiendo a distintas ideas o conceptos innovadores para convertirlas en comunidades verdes. En el artículo "*A multi-agent system approach to exploit demand-side flexibility in an energy community*" [3], publicado en la revista electrónica "*Science Direct*" y con la que colabora la UAH, propone el desarrollo de un sistema multiagente para modelar una comunidad energética, con miembros tanto residenciales como no residenciales, con diferentes roles, objetivos y preferencias. En él se deja constancia de la importancia de la estrategia de la gestión de la demanda en una comunidad energética para maximizar la autosuficiencia.

A su vez, otro escrito titulado "*Energetic communities for community energy: A review of key issues and trends shaping integrated community energy systems*" [4], presenta el concepto de "sistemas energéticos comunitarios integrados" como un desarrollo de futuro para reorganizar los sistemas energéticos locales con el fin de integrar los recursos energéticos distribuidos e involucrar a las propias comunidades. Estos sistemas pretenden garantizar un autoabastecimiento de energía y ampliar los servicios esenciales al sistema energético.

Otra interesante presentación titulada "*Flexibility in Multi-Energy Communities With Electrical and Thermal Storage: A Stochastic, Robust Approach for Multi-Service Demand Response*" [5], presenta un programa entero lineal mixto de energía/reserva para un sistema comunitario que cuenta que cuenta con restricciones en la red local. En este proyecto se presentan métodos para almacenar energía para posteriormente maximizar sus servicios aprovechando su trabajo, por ejemplo, con bombas de calor.

También se han buscado construcciones ya existentes catalogadas como verdes o sostenibles. El Consejo de la Construcción Ecológica de Estados Unidos (U.S. Green Building Council) es una organización dedicada a promover la industria de la construcción de edificios verdes. Rick Fredrizzi, presidente y fundador de esta organización, ha publicado numerosos proyectos relacionados con edificios verdes y sostenibles, entre los que se encuentra una guía de conceptos básicos de este tipo de edificios. En esta guía, aparte de recalcar con detalles el impacto de estas construcciones en el medioambiente, también habla cómo debe ser el pensamiento sustentable con este tipo de edificios: "Al diseñar edificios y comunidades, debemos comprender los elementos individuales del sistema y sus relaciones entre sí como un todo. Una sola decisión puede producir un efecto dominó. Por ejemplo, las mejoras en la envolvente de un edificio, el límite entre los elementos del interior y el exterior de un edificio, pueden cambiar los requisitos del sistema mecánico. El uso de un mejor aislamiento o de ventanas más eficientes puede permitir el uso de un sistema de calefacción más pequeño" [6].

Además, se han buscado ejemplos de edificios ecológicos ya existentes, del cual ha llamado especialmente la atención el edificio de oficinas "*Bloogmerg*" de Londres, mostrado en la Figura 5. Este edificio, construido en 2017 y puntero en el mundo en cuanto a sostenibilidad, incluye paneles de techo integrados con más de 500.000 leds, logrando un ahorro de energía de hasta un 40% con respecto a los fluorescentes tradicionales. Otra de sus peculiaridades es la conservación de agua, ya que se utiliza el agua residual (la lluvia de tejado, agua de soplado de la torre de refrigeración, lavabos, duchas, etc.) para servir a los inodoros de vacío, no utilizando así agua de red para sus descargas y propiciando un ahorro de 25 millones de litros de agua al año. Pero principalmente se ha podido observar la importancia que tiene el ajuste horario de la demanda gracias a un flujo de aire inteligente, el cual distribuye el CO_2 en función del número de personas que ocupan cada zona del edificio en un momento dado. Gracias al ajuste de dicho flujo en respuesta a las horas y patrones de ocupación se espera que se calcula un ahorro anual de entre 600 y 750 MWhr de energía [7].

Figura 5. Edificio verde "*Bloogmerg*" de Londres.

Un innovador artículo titulado "*Exploiting design thinking to improve energy efficiency of building*", también de "*Science Direct*", propone un enfoque de innovación en el diseño de energía inteligente para desarrollar un sistema de energía eficiente para edificios: "las fases que se han de seguir para alcanzar las innovaciones energéticas: descubrir, definir, desarrollar y entregar" [8]. Con este método, pretende proporcionar una base para innovar nuevas soluciones centradas en el consumidor para las oportunidades del sistema energético.

Para la construcción del montaje se han buscado proyectos basados en automatización de maquetas, donde se ha encontrado un proyecto de Joe Hanson, ingeniero de California, con una idea muy similar a la que se pretende hacer en este proyecto ya que además de automatización, utiliza juegos de piezas lego. Dicha maqueta consiste en una casa con 7 leds integrados representando la estufa o la chimenea entre otras cosas, y un motor paso a paso para abrir y cerrar una puerta, como se puede observar en la Figura 6 [9]. Además, incluye un blog en el que hay numerosos proyectos de automatización de procesos, muchos de ellos basados en robótica.

Figura 6. Ejemplo de casa Lego.

Para obtener información y facilitar el trabajo de la automatización mediante RPI, se han buscado blogs y trabajos relacionados con ella. Al ser una placa de bajo coste

vendida en todo el mundo, existen una gran cantidad de trabajos relacionados con ella y muy variados, pero se ha enfocado la búsqueda en proyectos basados en monitorización ya que, para ello, también se necesitan, entre otras cosas, controlar pines GPIO, diseño de *webservers* o cómo establecer las conexiones con ellas [10-13].

1.4- Estructura del documento

Para abordar este trabajo de forma ordenada, se definirán los siguientes puntos:

- **Capítulo 2. Diseño de la comunidad inteligente:** Se explicarán casos de uso y roles, se pondrán ejemplos de consumo y cómo se pueden controlar, las decisiones de montaje de la comunidad y el montaje final.
- **Capítulo 3. Escenarios de una comunidad inteligente:** Se verá qué tipo de escenarios se pueden cargar atendiendo a su consumo, las diferentes formas de cargar las programaciones horarias y las validaciones en el montaje.
- **Capítulo 4. Conclusiones y trabajo futuro:** Se resumirán las conclusiones obtenidas y se propondrán mejoras para el futuro derivadas del trabajo que se haya realizado.

2- Diseño de comunidad inteligente

2.1- Casos de usos y roles

Para comprender el funcionamiento de este proyecto, se va a proponer un caso particular de la calefacción de un hogar en invierno. Se desea que el hogar mantenga una temperatura entre 20 y 25 grados durante todo el día. Para ello se tendrá que mantener encendida un total de 8 horas diarias. Además de la calefacción, el consumidor desea activar la alarma entre las 8 y las 16 horas ya que en ese intervalo no se encontrará en el hogar.

Este consumidor, dispuesto a colaborar en la búsqueda de los objetivos de la comunidad sostenible, establece una cantidad de demanda fija para la alarma, así como un consumo variable para la calefacción. Esta última también se considera flexible ya que, aunque no sea sobre un horario fijo, el consumidor también puede tener preferencia por un determinado intervalo de tiempo. Para este caso, se considerará que no tiene ningún preferente en cuanto a la calefacción.

El consumidor pasará esta información a un sistema centralizado llamado agregador (del cual se hablará más en detalle a continuación), el cual es el encargado de hablar con los servicios públicos de suministro para asignar el suministro de electricidad esperado a través de un conjunto de fuentes renovables.

Dado que para la calefacción se tiene flexibilidad horaria, se puede elegir el horario que mejor le convenga, ahora bien ¿cuáles son las horas ideales para ponerla? En la Figura 7 se puede observar una gráfica obtenida de la app de la Red Eléctrica de la generación de energía renovable en España [14] en un día de invierno (por simplificar el ejemplo no se han tenido en cuenta las emisiones de CO_2, pero para casos reales se deberán tener en cuenta y encontrar la balanza más óptima para encontrar las mejores horas).

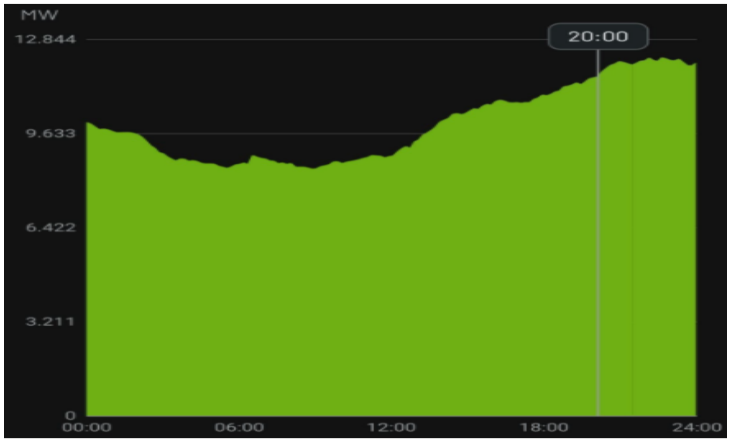


Figura 7. Media de consumo diario de energía eólica en España en invierno.

A la vista de la gráfica, las mejores horas para programar la calefacción serán entre las 16y las 24 horas ya que es cuando más energía verde se produce, por lo que la programación horaria quedará como se observa en la Tabla 1.

HORA	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Calefacción	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
Alarma	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0

Tabla 1. Programación horaria de la calefacción y de la alarma.

Esta programación pasa a un sistema centralizado con sistemas de agregación.

En resumen, para establecer la programación horaria, hay tres actores principales:

- **El consumidor**: Proporciona el uso de energía en el hogar para que sea controlada y gestionada automáticamente. En este ejemplo, será el que haga la programación horaria del aire acondicionado y de la calefacción.
- **Comunidad de propietarios:** En ella se encuentra un agregador de flexibilidad de la demanda el cual ayudará a optimizar la demanda conjunta de energía. En otras palabras, permite la distribución local de energía demandada por el consumidor. En el siguiente punto se hablará más en detalle sobre él.
- **Servicios públicos de energía:** Conjunto de proveedores de energía compartido por los clientes. Permite recolectar energía de fuentes renovables dirigidas a dar menores impactos ambientales e incrementar la seguridad del suministro.

Una vez que se conocen los actores principales, se plantean dos cuestiones: ¿Cómo se consigue que la programación horaria se ejecute en la calefacción y la alarma en este proyecto en concreto? ¿Qué papel juega cada uno de los actores?

Para responder a estas preguntas, se distinguen 4 pasos:

1- Se establece la programación horaria. Esta programación se hará con la ayuda de una página web en donde la RPI será el servidor web ya que, como se ha mencionado anteriormente, no solo es un microcontrolador, sino también un ordenador que permite conectarse a la red. Además, será necesario colocar más de una RPI, donde una será la principal, encargada de ejecutar el *webserver* y de leer e interpretar la programación horaria, y el resto las secundarias, que recibirán la información de la principal.

2- Si la programación horaria es correcta, el servidor web manda esta información a una base de datos dentro de la propia RPI.

3- Un script lee la base de datos y comprueba si los elementos seleccionados forman parte de la propia RPI (principal) o de alguna de las otras (secundarias). Si forma parte de la propia, activa los pines GPIO correspondientes en su horario correspondiente, si no, manda la programación de dichos elementos a las RPI secundarias para que activen sus elementos.

4- Los elementos, directamente conectados a los pines GPIO, se activan al activarse los pines GPIO. Pero para que todo esto se lleve a cabo, es necesario que los servicios públicos proporcionen la energía necesaria.

Pueden verse estos pasos de manera gráfica en el diagrama mostrado en la Figura 8.

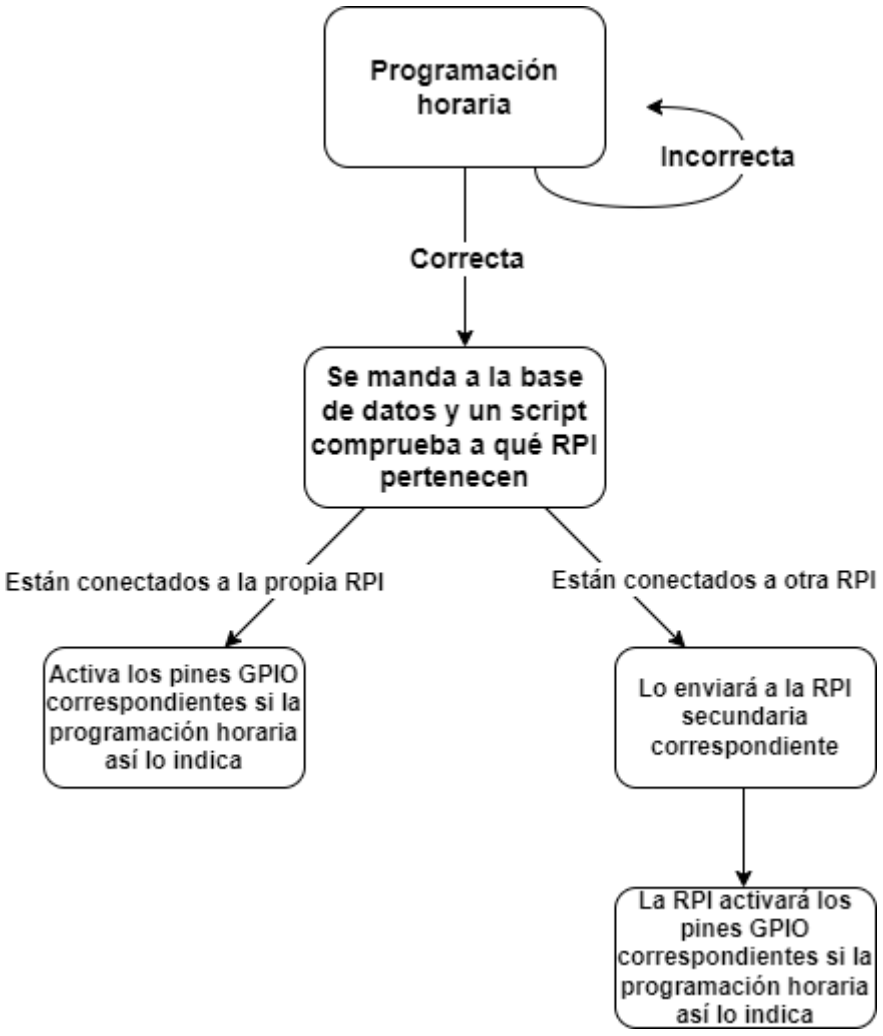


Figura 8. Diagrama de pasos que sigue la programación horaria hasta la RPI.

Y una vez que se saben los pasos, se verá cómo se ha diseñado la parte software de cada uno de ellos.

2.2- Diseño del *webserver*

La interfaz web pretende ser sencilla y fácil de entender para el usuario [15]. Para ello, se pone el diseño final del edificio pudiéndose acceder a cada casa por separado simplemente clicando en ella para ver sus elementos y establecer su programación manualmente. También está la opción de hacerlo de manera automática con una carga global de todos los elementos a través de un fichero CSV. Los tipos de carga se explicarán más adelante. El diseño de la página es el mostrado en la Figura 9.

Figura 9. Diseño del *webserver*.

Además, como se observa, incluye la hora actual, que se irá actualizando automáticamente. Si se clica, por ejemplo, en la cafetería, se verá su plano con cada elemento del que dispone, y si se clica en uno de estos elementos, se podrá hacer su programación simplemente activando o desactivando el interruptor de la hora correspondiente. Una vez que se tenga la programación deseada, se establecerá pulsando el botón "*Set programación*" para que quede registrada. En la Figura 10 se puede ver el plano de la cafetería con la leyenda de símbolos.

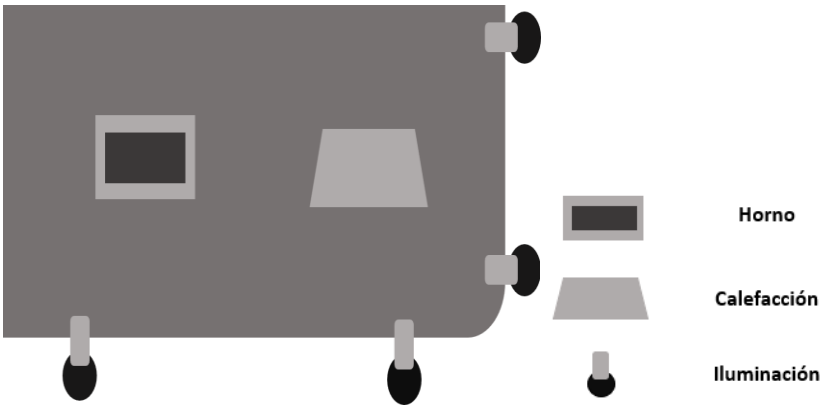


Figura 10. Plano de la cafetería y su leyenda de símbolos.

Pero antes de acceder a estas páginas web, el administrador deberá introducir las credenciales por seguridad desde la página de inicio, mostrada en la Figura 11.

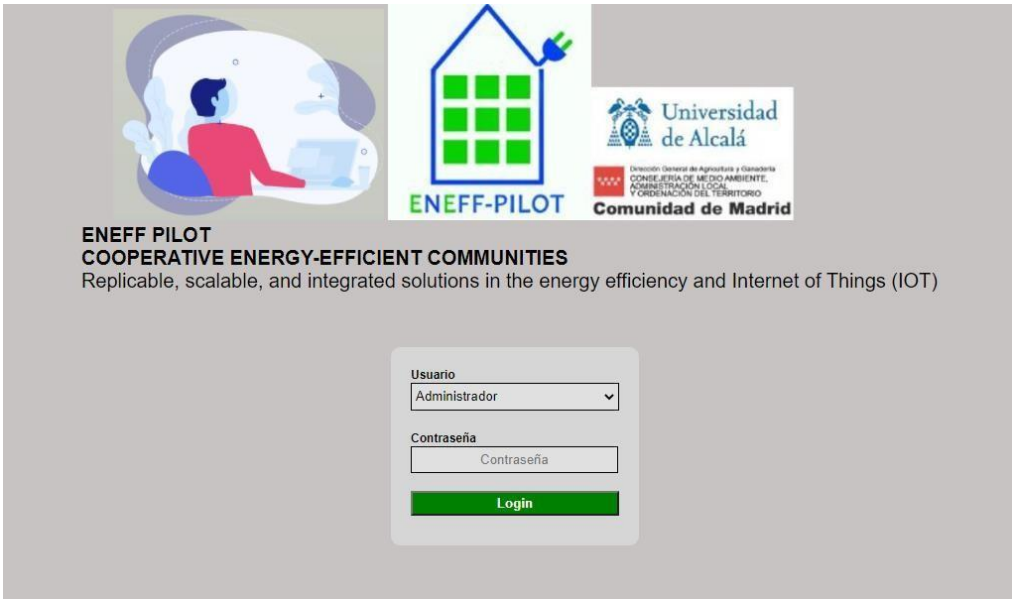


Figura 11. Página de inicio de sesión.

Para la implementación de este *webserver*, se ha programado en tres lenguajes combinados entre sí:

- **HTML:** se utiliza para estructurar y desplegar una página web y sus contenidos. El navegador desde el que se establezca la conexión será el que interprete este lenguaje.
- **SQL:** Es un lenguaje diseñado para administrar la información de la base de datos. Se utilizará para leer y depositar la información deseada.

- **PHP:** Hipertext Preprocessor, procesador de hipertexto en inglés. Facilita la conexión entre los servidores y el interfaz del usuario a través de la base de datos.

Además, las bases de datos se administrarán a través de *Mysql* [16], que es un software diseñado para tal fin y de código abierto. Con esta base de datos, además de almacenar la programación horaria que se mandarán a los pines GPIO será la encargada de comunicar las distintas RPIs entre sí.

En los Anexos se muestra el código realizado de la página principal, pero aquí se van a ver las partes más importantes de ella (Figura 9) con el pseudocódigo mostrado a continuación:

Pseudocódigo *home.php*

Cabecera: Título de la página web (Comunidad Sostenible)
Archivos de cabecera (equivalentes a ficheros .h en c)
Tipo y tamaño de letra
Definición de variables y funciones
Lectura de la hora en la base de datos cada segundo
Timeout de 5 minutos en caso de inactividad

Cuerpo: Color de fondo
Título de la página (Comunidad con electricidad agregada)
Subtítulo dinámico de la página
Carga de la Imagen de la comunidad
Definición de botones y ruta a la que se dirige

Algoritmo 1. Pseudocódigo de la página principal del *webserver*

A partir de esta página saldrán las páginas secundarias, es decir, la de los establecimientos. En el algoritmo 2 se muestra el pseudocódigo de la página de la cafetería ("cafetería.php"), pero el resto de las páginas se realiza de forma análoga.

Pseudocódigo cafetería.php

Cabecera: Título de la página web (Comunidad Sostenible)
Archivos de cabecera (equivalentes a ficheros .h en c)
Tipo y tamaño de letra
Definición de variables y funciones
Timeout de 5 minutos en caso de inactividad

Cuerpo: Color de fondo
Título de la página (Comunidad con electricidad agregada)
Subtítulo
Carga del plano de la cafetería
Definición de botones con sus respectivas etiquetas
Botones de la programación manual horaria (ocultos hasta que se soliciten)

Algoritmo 2. Pseudocódigo de la página de la cafetería

Para acceder a este *webserver* se hará mediante una conexión de área local con conexión Ethernet. Cada una de las RPIs se conectarán a un conmutador con direcciones IP dentro de una misma subred para que sea posible la comunicación entre ellas, y del conmutador se deberá establecer la conexión al host con el que se desea establecer la conexión. Para visualizar la página, bastará con poner la dirección IP de la RPI principal en el navegador.

2.3- Controlador de consumo

En este proyecto se demuestra cómo los consumidores adaptan su consumo de energía de forma cooperativa y centralizada. Comparten su programa de demanda con un recopilador de datos llamado **Agregador de demanda eléctrica planificada.**

El concepto o acto de agregación puede definirse como "la agrupación de diferentes clientes dentro del sistema eléctrico para que actúen como una sola entidad cuando participen en los mercados de electricidad o cuando vendan servicios a los operadores del sistema" [17]. Se podría decir que la agregación de demanda es la agrupación de diferentes clientes pequeños dentro del sistema eléctrico (comercializadores, distribuidores y productores, etc.) para que participen en los mercados de electricidad de manera conjunta para lograr un beneficio común. Esto no se puede lograr sin cierta solidaridad por parte de los consumidores ya que requiere asumir que en determinados momentos han de variar los consumos.

El agregador facilita la información de consumo de energía y establece una asignación de recursos optimizados respondiendo a las condiciones de suministro, apuntando a fuentes renovables.

En este proyecto, cada cliente sería cada vecino de la comunidad, y las RPI serían las encargadas de agrupar la información de la comunidad para gestionarla de manera eficiente. Aunque difieren con un agregador en que hay más de una tarjeta (no es solo un dispositivo) y se interactúa con ellas mediante una página web y no por una aplicación inteligente mandando órdenes directas, se ha decidido hacer de esta forma ya que el concepto se muestra de manera clara.

2.4- Montaje

Diseño

El primer paso del montaje es el diseño de la comunidad inteligente. Tras adquirir los juegos de piezas Lego (mostrados anteriormente en la Figura 1), la distribución de establecimientos se construirá como se muestra en la Figura 12.

Figura 12. Esquema de la distribución de los establecimientos.

Ambos legos se unirán en uno solo, formando una sola comunidad. Éstos se apoyarán sobre una base la cual deberá incluir también las RPI, el cableado y las placas *protoboard*. Las dimensiones del primer Lego son de 38x30x25 cm (base, altura, profundo), y las del segundo de 25x25x20 cm. La base deberá incluir, además de los legos, las RPI, las *protoboard* que facilitarán el cableado y los propios cables, por lo que se prevé que la base será al menos de 100x50 cm. La distribución inicial pensada vista desde arriba es la mostrada en la Figura 13.

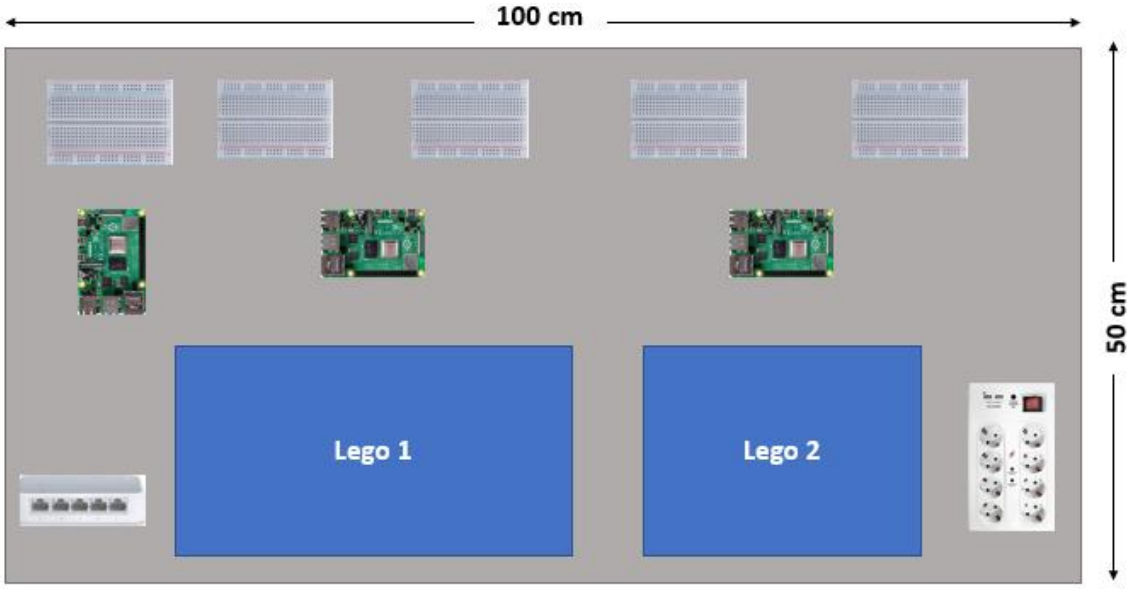


Figura 13.Esquema de la distribución de los elementos.

Como se puede ver, en la parte trasera irán incluidas las RPI junto a las *protoboards* y los cableados, lo cual se tratará de tapar en la medida de lo posible. Cada una de las RPI tendrán las funciones mostradas en la Figura 14.

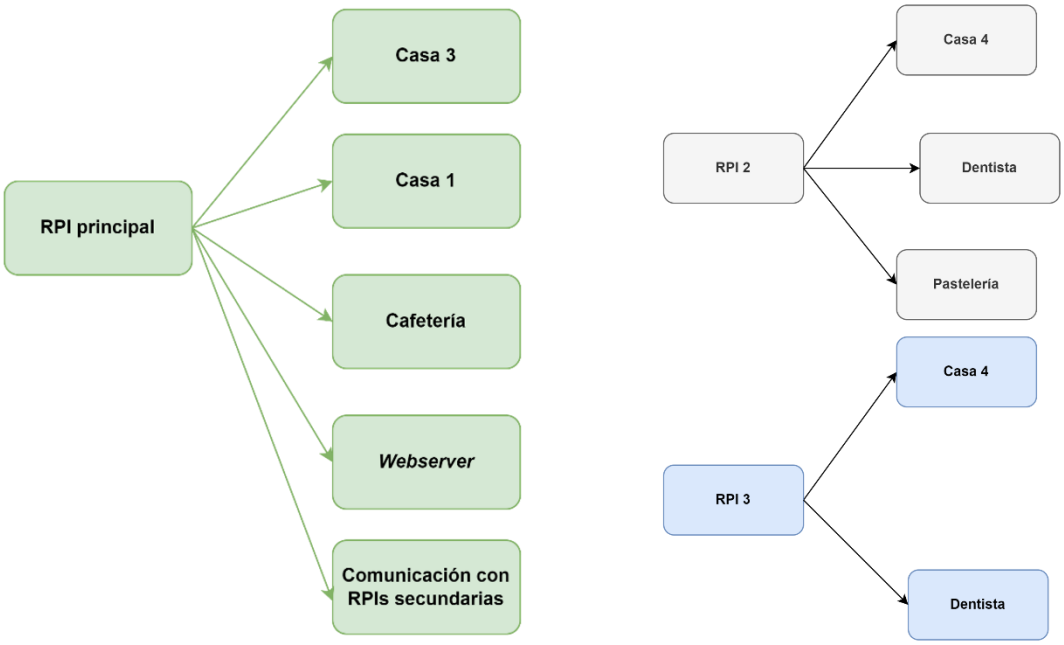


Figura 14. Funciones de cada RPI.

Por delante, estarán ambos Legos unidos formando una sola comunidad, y a los extremos estarán el conmutador que conectará mediante cable Ethernet todas las RPI y la regleta de enchufes que permitirá alimentarlas. El cableado, que irá desde las RPI a las *protoboards* y a los Legos, se tratará de que vaya pegado al suelo con pegamento termofusible en la medida de los posible para evitar el desorden de cables. Además,

para facilitar el cableado desde los pines de las RPI en las *protoboard*, se incluirán en estas últimas los adaptadores mostrados en la Figura 15, el cual irá conectado mediante un cable plano hasta la propia RPI.

Figura 15. Adaptador para la RPI.

Construcción

Se comienza construyendo el primer Lego siguiendo el manual incluido. La parte exterior apenas se modificará con respecto a las instrucciones incluidas, haciendo ligeros cambios en el interior para incluir los elementos. La ventaja que presentan es que se puede construir cada establecimiento por separado y posteriormente unirlos todos, por lo que la idea inicial es hacerlos todos por separado, incluir los componentes electrónicos y por último unirlos todos.

El primer establecimiento para construir es la cafetería. Los elementos que se incluirán en ella serán calefacción, horno e iluminación exterior. Siguiendo los pasos del manual, el resultado es el mostrado en la Figura 16.

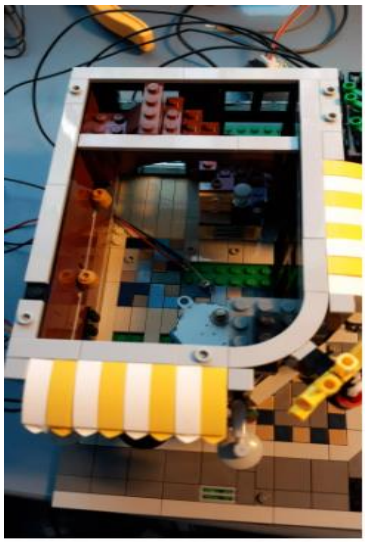


Figura 16. Construcción de la cafetería.

Como se observa, en el interior no se han incluido ciertos elementos con el fin de poder incluir con facilidad los elementos y que sean visibles desde fuera.

Tras construirse la cafetería, se seguirán los mismos pasos con la pastelería y el taller (este último del segundo Lego), que formarán el piso 0 de la comunidad. La unión de ambos Legos se hace de forma sencilla ya que las piezas de ambos Legos son del mismo tamaño y grosor, por lo que se pueden combinar entre ellas. Bastará con poner piezas que junten ambas bases, quedando unidas como una sola pieza.

Una vez construido el piso 0 y unido entre sí, como se observa en la Figura 17 se procede a construir la base mencionada anteriormente. Además de incluir todos los elementos mencionados anteriormente, deberá ser robusta ya que sobre ella se apoyarán todos los elementos.

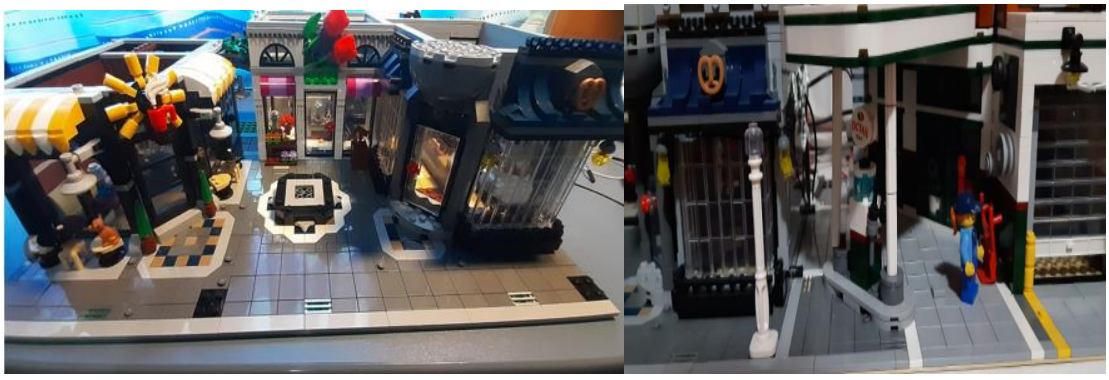


Figura 17. Construcción del piso 0 y unión de ambos Legos.

El grosor de la base será de 2 cm aproximadamente, de manera que sea capaz de soportar todo el peso de los Legos y los elementos incluidos. Una vez que se tiene hecha

la base, se fijan los Legos a ella junto a las placas *protoboard* y las RPI y se procede a incluir los primeros elementos electrónicos.

El primer elemento para incluir será la iluminación externa de la cafetería. Para ello, se pondrán diodos led de color blanco en las farolas que incluye el propio Lego. La RPI entrega un nivel de 3.3V, y según el fabricante, estos diodos leds requerirán unos 3V aproximadamente, por lo que habrá que colocar una pequeña resistencia (de 20 ohm) además del cableado para evitar posibles sobrecargas. El cableado se tratará de llevar pegado a la pared, siendo del mismo color que esta, para que se vea lo mínimo posible. En la Figura 18 dicho cableado provenientes de los diodos led.

Figura 18. Iluminación externa de la cafetería.

De manera similar se incluyen el resto de los elementos en el piso 0. La distribución de elementos electrónicos queda como se muestra en la Tabla 1.

Establecimiento	Elemento 1 (Dispositivo electrónico)	Elemento 2 (Dispositivo electrónico)	Elemento 3 (Dispositivo electrónico)	Elemento 4 (Dispositivo electrónico)
Cafetería	Iluminación (Leds blancos)	Calefacción (Leds naranjas)	Horno (Led rojo)	-
Pastelería	Iluminación (Leds blancos)	Aire acondicionado (Motor byj48)	Riego automático (Led azul)	Horno (Led rojo)
Taller	Iluminación (Leds blancos)	Alarma (Leds rojos parpadeantes)	Aire acondicionado (Motor byj48)	Calefacción (Leds naranjas)

Tabla 1. Elementos incluidos en el piso 0.

Una vez cableados el piso 0, el aspecto que presenta es el mostrado en la Figura 19.

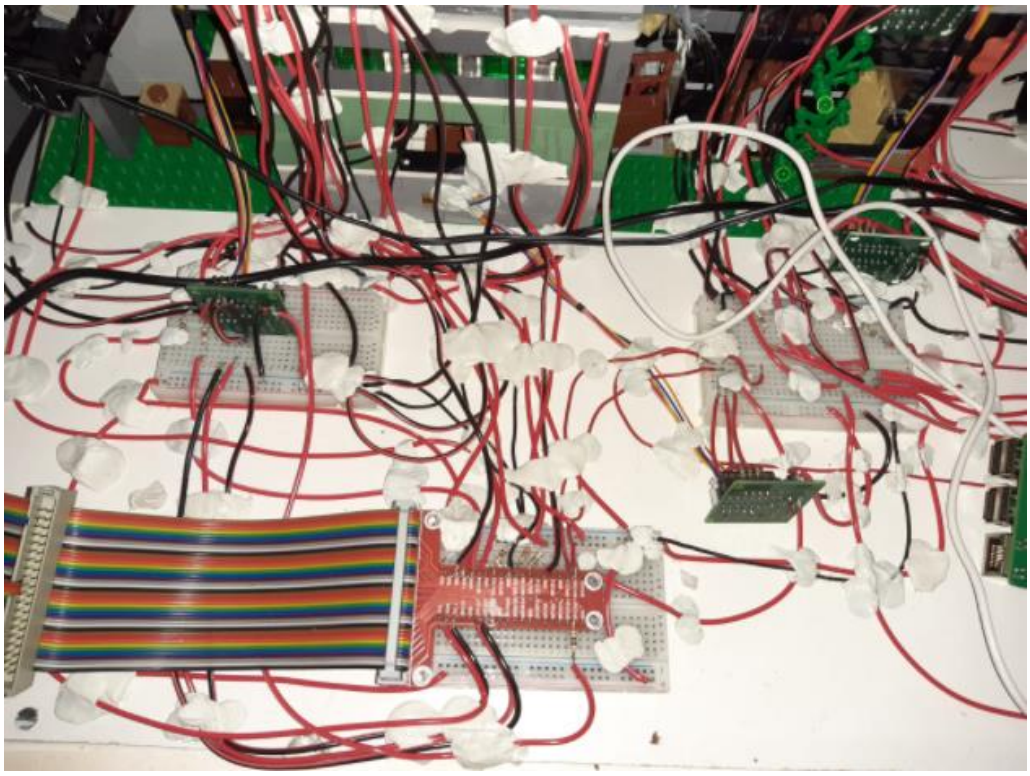


Figura 19. Cableado del piso 0.

Seguidamente, se procede a construir el piso 1. Al igual que con el piso 0, se construyen por separado para incluir los elementos más fácilmente. En este caso, al estar algo más elevados, el cableado deberá ser más largo, por lo que para evitar que se haga todavía más engorroso, cada casa se llevará a los pines GPIO de una RPI distinta, optimizando así el espacio. Además de los elementos a incluir en las propias casas, habrá que incluir la iluminación del piso inferior, ya que el suelo de este piso será el techo del inferior. Esta iluminación no será un elemento programable, sino que se considerará un elemento fijo para que la visión en el interior de las casas sea siempre la óptima. Los elementos de este piso son los mostrados en la Tabla 2.

Establecimiento	Elemento 1 (Dispositivo electrónico)	Elemento 2 (Dispositivo electrónico)	Elemento 3 (Dispositivo electrónico)	Elemento 4 (Dispositivo electrónico)
Casa 1	Calefacción (Leds naranjas)	Lavadora (Motor byj48)	Riego (Led azul)	-
Dentista	Proyector (Led multicolor)	Iluminación (Leds blancos)	Aire acondicionado (Motor byj48)	-
Casa 2	Riego automático (Leds azules)	Calefacción (Leds naranjas)	Aire acondicionado (Motor byj48)	Lavadora (Motor byj48)

Tabla 2. Elementos incluidos en el piso 1.

De manera muy similar, se construye también el piso 2 (casa 3 y casa 4). Este piso irá construido en el Lego 1 ya que el Lego 2 no incluye segundo piso, lo que facilitará de manera significativa el cableado ya que desde esa altura se requerirán cables de

longitudes de hasta 50 cm en algunas ocasiones. La distribución de elementos de este piso son los mostrados en la Tabla 3.

Establecimiento	Elemento 1 (Dispositivo electrónico)	Elemento 2 (Dispositivo electrónico)	Elemento 3 (Dispositivo electrónico)	Elemento 4 (Dispositivo electrónico)
Casa 3	Calefacción (Leds naranjas)	Lavaplatos (Led azul)	Aire acondicionado (Motor byj48)	-
Casa 4	Riego automático (Led azules)	Horno (Led rojo)	Lavaplatos (Led azul)	-

Tabla 3. Elementos incluidos en el piso 2.

Tras la inclusión de ambos pisos, se incluye también el conmutador y la regleta de enchufes para conectar las RPI. El diseño final visto de frente es el mostrado en la Figura 20.

Figura 20. Imagen del Lego completo iluminado.

Como se observa, el cableado apenas se puede ver en la vista frontal, que es desde donde se hará la presentación del proyecto. En la Figura 21 se puede ver la parte trasera, donde se incluye todo el cableado, las RPI con sus adaptadores y las placas *protoboard*.

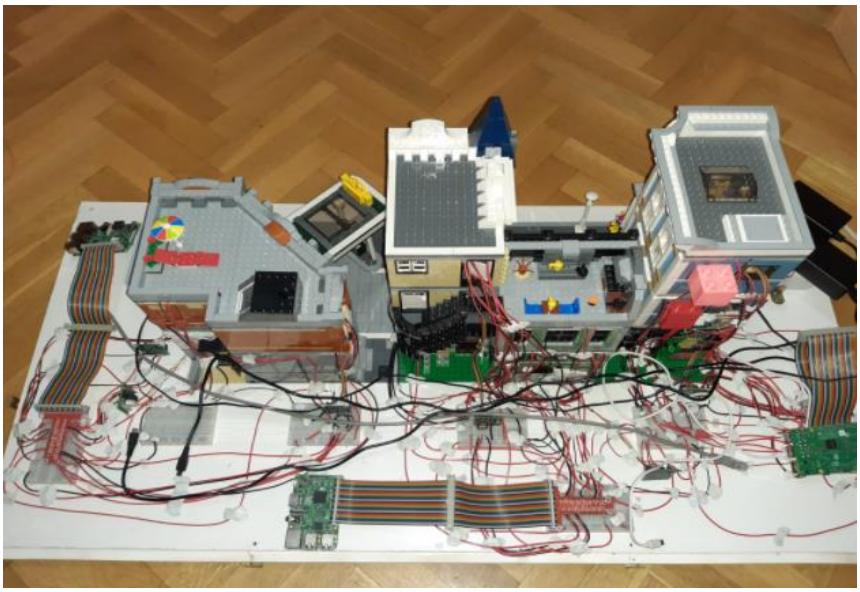


Figura 21. Imagen trasera del Lego visto desde arriba.

Finalmente, para finalizar el montaje, esta parte se ocultará situando un cajón hueco que se podrá quitar y poner fijándose mediante unos enganches situados en los extremos de la parte trasera de la base. En la Figura 22 se puede volver a ver la parte trasera del Lego con el cajón ocultando el cableado.

Figura 22. Imagen del Lego visto desde arriba.

En total se han incluido veintisiete elementos, siete de ellos motores. Dado que para cada motor se han utilizado 4 pines GPIO, uno por cada bobina, han sido necesarios cuarenta y ocho pines GPIO. El motivo de usar tres RPI en lugar de dos (ya que entre dos suman 48) ha sido evitar el excesivo consumo que provocaría el caso de tener todos sus pines activos a la vez, además de dificultar la optimización de espacio y cableado en la base.

Comunicaciones

Una vez hecho el montaje, se va a explicar cómo se ha programado el script "*controlador.c*" que controla estos componentes. Este script se hará en las tres RPIs.

El primer paso es instalar en las RPI las librerías necesarias para su desarrollo. El lenguaje de programación de este script será en C, aunque no será necesario instalar las librerías propias de dicho lenguaje, como "*stdio.h*", ya que vienen instaladas por defecto, por lo tanto, solo serán necesarias dos, la encargada de leer la base de datos, llamada "*mysql.h*", y la encargada de controlar los pines GPIO, denominada "*BCM2835.h*".

El sistema operativo de la RPI es "*Raspbian*", que es una distribución de Linux, por lo que su funcionamiento es muy similar. Para instalar estas librerías simplemente se utiliza el comando "*sudo apt-get install*" seguido del nombre de la librería. En caso de que se haya instalado sin problemas, simplemente habrá que situarla en la misma carpeta donde se realiza el script y ya estarán listas para usarse.

Estos pasos se han de hacer en las tres RPIs ya que cada una tendrá un script con sus propios componentes. En el Algoritmo 3 se puede ver el pseudocódigo del script programado en cada una de ellas.

Pseudocódigo controlador.c

1- Inicio
2- Definición de variables
3- Definición de pines GPIO
4- Solicitar conexión con la base de datos
5- En caso de ser correcta la conexión, solicitar la hora actual y la programación horaria, en caso contrario, volver a solicitar hasta que haya una respuesta correcta
6- Comprobar la programación de cada elemento en su hora correspondiente. Si está a 1, activarlo, si es un 0, desactivarlo
7- Si no ha habido ningún error, volver al punto 4, en caso contrario, salir del programa

Algoritmo 3. Pseudocódigo del script "controlador.c"

Como se puede ver en este pseudocódigo, en el punto 6 se activan o desactivan los elementos electrónicos. En el caso de que el aparato electrónico esté representado por un motor, será necesario estar poniendo a nivel alto y bajo continuamente sus pines GPIO, ahora bien, ¿cómo se hace esto mientras el script continúa ejecutándose?

Para ello, se realiza un script secundario que se estará ejecutando de forma paralela y se encargará únicamente de mantener en funcionamiento los motores o desactivarlos cuando corresponda. El momento de activación o desactivación de lo dirá el script principal a través de la base de datos. El pseudocódigo de este script ("controlador_motores.c") es el mostrado en el Algoritmo 4.

Pseudocódigo controlador.c

1- Inicio
2- Definición de variables
3- Definición de pines GPIO
4- Solicitar conexión con la base de datos
5- En caso de ser correcta la conexión, comprobar qué motores se tienen que activar
6- Si se tiene que activar alguno de ellos, llamar a su función en un bucle de 7 pasos, uno por cada caso:
   - Caso 1: Las 3 bobinas a nivel bajo
   - Caso 2: Bobina 1 a nivel alto
   - Caso 3: Bobina 1 y 2 a nivel alto
   - Caso 4: Bobina 2 a nivel alto
   - Caso 5: Bobina 2 y 3 a nivel alto
   - Caso 6: Bobina 3 a nivel alto
   - Caso 7: Bobina 3 y 1 a nivel alto

   Fin
7- Si no ha habido ningún error, volver al punto 4, en caso contrario, salir del programa

Algoritmo 4. Pseudocódigo del script "controlador_motores.c"

Este código se ejecutará tiene un tiempo de ejecución aproximado de 50 ms, por lo que el retardo entre que se llama a la función de activación de motores y se comprueba si debe seguir activo es imperceptible.

Una vez realizados todos los scripts, surge una última cuestión: ¿Habrá que ejecutar dichos scripts cada vez que se conecten las RPI? La respuesta es que no. El sistema operativo de Raspbian cuenta con una herramienta denominada "*Crontab*" que permite programar una tarea en el momento deseado, el cual puede ser al iniciarse la RPI mediante la instrucción "*@reboot*", como se muestra en la Figura 23.

```
@reboot /eclipse/controlador&
@reboot /eclipse/controlador_motores&
```

Figura 23. Programación de la herramienta "*Crontab*".

3- Escenarios de comunidad inteligente

Para conseguir que una comunidad sea más sostenible y verde, es indispensable la colaboración de los consumidores. En la actualidad, los niveles de participación del consumidor todavía se encuentran en niveles de desarrollo. Para conseguir que esta participación suba, se ha de analizar el comportamiento de cada tipo de consumidor que participa en la programación de demanda agregada y clasificarse en distintos escenarios.

3.1- Tipos de escenarios

Para elaborar los distintos escenarios en los que se pueden clasificar los tipos de consumos de una comunidad sostenible, se han de tener en cuenta los siguientes factores:

1- **Tamaño de la comunidad:** Dependiendo del número de consumidores, se podrá clasificar en una comunidad más grande o más pequeña. En el demostrador hecho para este proyecto, se considerará una comunidad pequeña ya que solo tiene 9 consumidores. Para considerar una comunidad grande, se debería al menos tener más de 30 participantes en la demanda.

2- **Número de aparatos:** El número total de aparatos que debe programar el agregador.

3- **Volumen de la demanda:** El volumen de la demanda desplazable (es decir, la que es flexible a adaptarse a ciertos horarios) repercute en el rendimiento y depende del volumen y el flujo de la disponibilidad de energías renovables.

4- **Flexibilidad de los consumidores:** La elasticidad de la demanda de los consumidores influye directamente en la sostenibilidad de una comunidad. Cuanto más flexibles sean sus consumidores, mayor beneficio se obtendrá.

5- **Flujo de suministro:** La disponibilidad de generadores de energía renovable.

Un experimento basado en ML (*Machine learning*) realizado en el proyecto “Eneff-pilot”, analiza el comportamiento de una comunidad de 5 consumidores con 48 aparatos cada uno combinando estos factores [18].

En primer lugar, se simula una comunidad con una alta actividad y ocupación a lo largo del día. Las cargas de consumo se distribuyen de forma aleatoria.

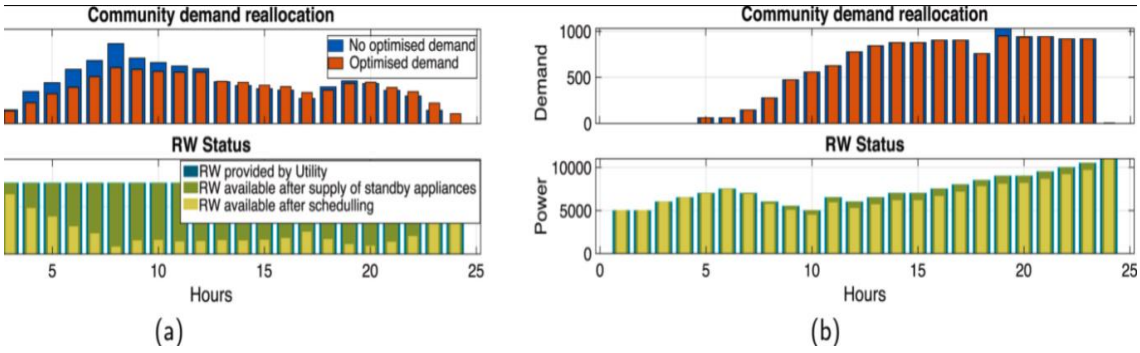


Figura 24. Simulación de la demanda de una comunidad poco flexible.

Como se observa en el caso a, las horas pico se alcanzan entre las 8 y las 11 de la mañana, siendo este periodo el más importante a la hora de suavizar la demanda. El agregador trata de distribuir las cargas a lo largo del día. Esta distribución se hace costosa al ser una comunidad con poca flexibilidad.

En el caso b, se tiene un escenario de una comunidad con poca actividad y ocupación durante el día, pero también con poca flexibilidad. Las cargas de consumo se distribuyen durante el tiempo libre, dificultando nuevamente la labor del agregador. En las gráficas inferiores, en color verde, se puede ver la energía renovable restante tras la programación horaria. En el caso b, apenas se puede aprovechar.

Se pasa ahora a un tipo de comunidad más flexible. Al igual que en el caso anterior, en el caso a de la Figura 25 se puede observar una comunidad con una alta actividad por parte de los miembros y además de una alta ocupación.

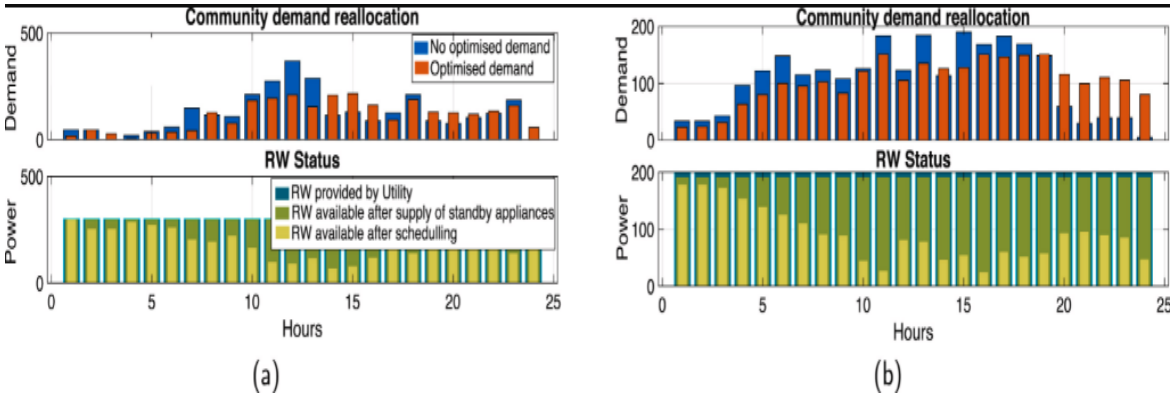


Figura 25. Simulación de la demanda de una comunidad consumidora semiflexible.

Como se observa, en el caso a se dan comportamientos de consumidores muy exigentes sobre todo en horas punta, y algo menos exigentes a lo largo del día en el caso b. Entre las 10 y las 11 de la mañana se alcanza el pico en el caso a, mientras que en el b se tienen picos distribuidos sueltos a lo largo del día. Se puede ver uno a las 11 horas, otro a las 15 y otro a las 18. En ambas gráficas se puede ver cómo el agregador consigue aplanar

la demanda en mayor medida en horas punta, incluso cuando la oferta es insuficiente en ciertas franjas horarias.

Por último, se simulan nuevamente dos casos, uno con mucha actividad y ocupación por parte de los consumidores (caso a) y otro con poca actividad y ocupación (caso b), pero en este caso los consumidores muestran gran flexibilidad, como se observa en la Figura 26.

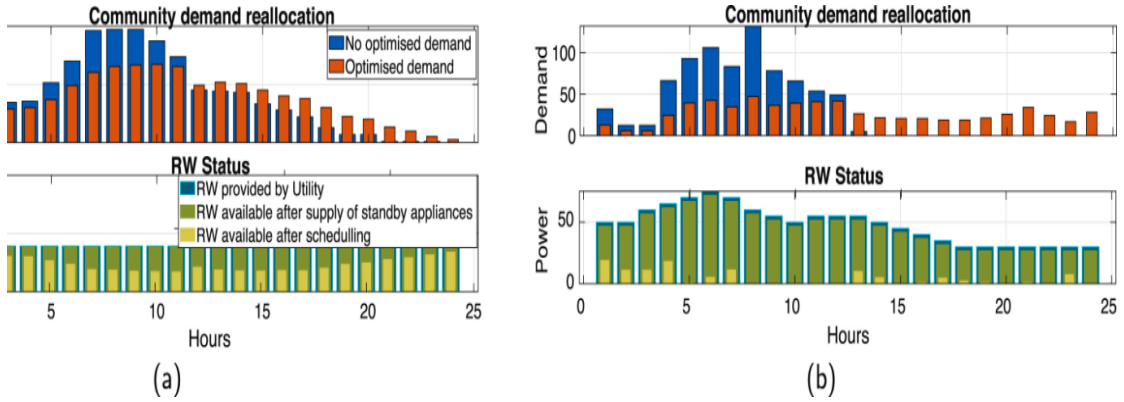


Figura 26. Simulación de la demanda de una comunidad consumidora flexible.

En el caso a, los consumidores demandan un consumo bajo en horas punta, con un suministro plano de renovables. En el caso b, a pesar de haber cierta falta de renovables en alguna hora del día, se consigue aplanar gracias a la flexibilidad de los ocupantes.

Así pues, según los comportamientos identificados, se pueden distinguir los siguientes escenarios:

1- **Busy Behaviour (Comunidad ocupada):** Los consumidores demandan mucho y crean un escenario difícil de ajustar cuando su flexibilidad es estricta y se concentra en los mismos periodos de tiempo. El agregador tiene poco margen para aplanar la demanda a lo largo del día y apenas consigue asignar la oferta disponible, como se muestra en la Figura 23. Las comunidades que presentan este patrón pueden presentar hasta un 3% del volumen de demanda flexible.

2- **Demanding behaviour (Comunidad demandante):** Estas comunidades muestran un amplio dinamismo de la flexibilidad de los consumidores. En la Figura 24 se puede observar este tipo de escenario. El agregador consigue aplanar el consumo en horas punta de forma eficiente, incluso cuando la demanda de renovables es insuficiente. Las comunidades que muestran este patrón transforman hasta un 15% del volumen de demanda flexible.

3- **Concerned behaviour (Comunidad preocupada):** Las comunidades que muestran un comportamiento de ahorro de energía se definen por ser

consumidores poco exigentes que demuestran una alta flexibilidad en la demanda a través de largos periodos de tiempo de preferencias. En la Figura 25, se ve cómo el agregador aplana la demanda máxima y hace coincidir la demanda de electricidad con los periodos en los que la energía renovable intermitente está disponible. Debido a la flexibilidad de los participantes, el tiempo de funcionamiento de las cargas variables puede desplazarse a distintas horas del día. Las comunidades que muestran este patrón disponen de hasta un 30% de flexibilidad en el volumen de la demanda.

3.2- Carga automática

Como se ha visto en el diseño del software, una de las formas que tiene el administrador de establecer la programación horaria en el agregador es de manera automática mediante un fichero CSV, el cual se puede descargar desde el propio software.

En este fichero, se puede establecer la programación horaria de todos los elementos electrónicos que conforman el montaje, siendo un 0 el estado apagado y un 1 el estado encendido, durante cada una de las 24 horas del día. En la Figura 27 se puede ver un ejemplo de este fichero.

1	Hora	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2																									
3	Cafeteria																								
4	Iluminacior	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	Calefaccic	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	Horno	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7																									
8	Pasteleria																								
9	Iluminacior	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
10	AC	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
11	Riego autc	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12	Horno	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	1	0	0	1	0	0
13																									
14	Taller																								
15	Iluminacior	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	AC	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
17	Alarma	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
18	Calefaccic	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1
19																									
20	Casa 1																								
21	Calefaccic	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
22	Lavadodra	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0
23	Riego	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
24																									
25	Dentista																								
26	Proyector	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
27	Iluminacior	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
28	AC	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1
29																									
30	Casa 2																								
31	Riego autc	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	1	0
32	Calefaccic	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
33	AC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
34	Lavadora	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0
35																									
36	Casa 3																								
37	Calefaccic	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	1
38	Lavaplatos	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	AC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40																									
41	Casa 4																								
42	Riego autc	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0
43	Lavaplatos	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
44	Horno	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

Figura 27. Ejemplo de fichero de carga automática en el agregador.

La intención de hacerla en formato CSV es la de facilitar al administrador su labor ya que es una plantilla sencilla e intuitiva. Una vez guardada la programación deseada, el administrador simplemente deberá volver a cargarla en el software y de manera automática y la programación será establecida en el montaje.

Además, desde el propio Excel, es muy sencillo sacar gráficas o diagramas de barras con la programación establecida de manera que el administrador podría ver las horas de máximo consumo y ver el grado de optimización del que dispone la comunidad. Podría sacarse, por ejemplo, un diagrama de barras con la suma total de horas de los elementos de cada establecimiento, como se muestra en la Figura 28.

Figura 28. Gráfico del total de horas de consumo por establecimiento.

Otro gráfico que se podría sacar para la optimización de la demanda sería la cantidad de elementos simultáneos que tiene en uso cada establecimiento por horas. En la Figura 29 se puede ver dicho gráfico del taller.

Figura 29. Gráfico de barras de elementos simultáneos utilizados en el taller para un determinado escenario.

3.3- Carga manual

El segundo método para que el administrador pueda realizar la programación horaria en el agregador es a través de una carga manual. Mediante este método, el administrador puede ir casa por casa visualizando mediante interruptores la programación que se encuentra actualmente establecida en cada uno de los elementos del establecimiento en el que se encuentre, y se pueden modificar según se considere. Este método está pensado para que se puedan realizar pequeños cambios a la programación existente o pueda consultar la programación actual de un elemento en concreto sin necesidad de descargar el fichero CSV.

En la Figura 30 se puede ver un ejemplo del horno de la cafetería en este tipo de carga. Los interruptores situados en la derecha en color verde marcan que a dichas horas estará encendido, mientras que en las restantes permanecerá apagado. Para cambiar su programación el alguna de las horas, bastará con pulsar uno de estos interruptores para cambiar su estado, y se mandará la programación al pulsar sobre el botón "*Set programación*".

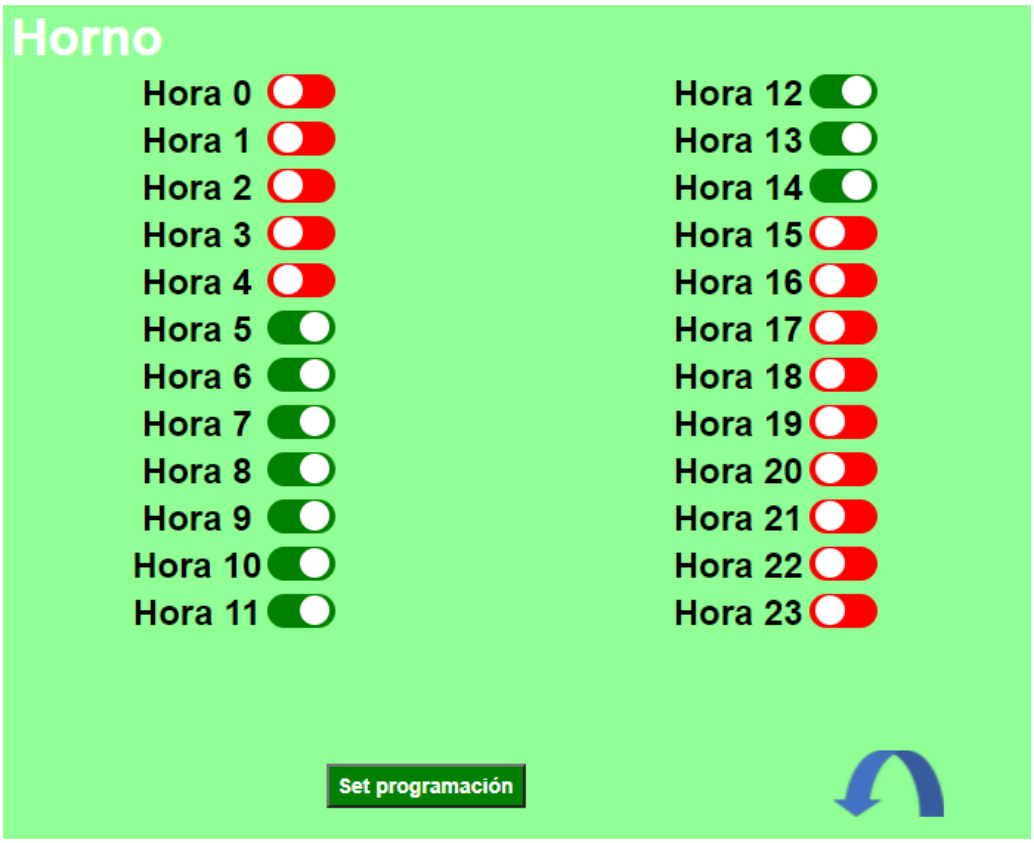


Figura 30. Ejemplo de carga manual del horno de la cafetería.

En este tipo de carga, se ha pretendido que sea visual y de fácil comprensión para el administrador. La carga se puede hacer con todos los elementos existentes de todos los establecimientos, pero también está pensada para que se pueda ver la programación actual de manera más visual que el fichero CSV, ya que este sólo incluye ceros y unos.

3.4- Validación en montaje

Una vez visto cómo se ha diseñado este montaje, cómo se ha montado y cómo funciona, se va a probar a hacer la validación en montaje. Para ello, se cargará un fichero CSV con cada tipo de escenario visto anteriormente y se comprobará su correcto funcionamiento.

En primer lugar, se carga un escenario *concerned*, es decir, con poco volumen, mostrado en la Figura 31.

1	Hora	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2																									
3	Cafeteria																								
4	Iluminacion	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	1	1	1	0	0	0	0	0
5	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
6	Horno	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1
7																									
8	Pasteleria																								
9	Iluminacion	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0
10	AC	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0
11	Riego auto	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0
12	Horno	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	0	0	0	0
13																									
14	Taller																								
15	Iluminacion	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	0	0	0	0	0	0
16	AC	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1	1	0	0	0	0	0
17	Alarma	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	0	0	0	0
18	Calefaccion	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0
19																									
20	Casa 1																								
21	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
22	Lavadodra	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
23	Riego	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
24																									
25	Dentista																								
26	Proyector	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1	1	0	0	0	0	0
27	Iluminacion	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
28	AC	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	1	1	1	1	1	0	0	0	0
29																									
30	Casa 2																								
31	Riego auto.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
32	Calefaccion	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0
33	AC	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	1	0	0	0	0	0	0	0
34	Lavadora	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
35																									
36	Casa 3																								
37	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
38	Lavaplatos	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
39	AC	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
40																									
41	Casa 4																								
42	Riego auto.	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1
43	Lavaplatos	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
44	Horno	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

Figura 31. Escenario *concerned* para validación del montaje.

Para ver de manera visual el número total de aparatos funcionando, se saca una gráfica de barras, mostrada en la Figura 32.

Figura 32. Gráfica del escenario *concerned* simulado en el montaje.

Como se observa en la gráfica, el volumen se distribuye a lo largo del día sin tener picos de consumo. Entre las 8 y las 10 de la mañana se alcanza el consumo mayor.

Para comprobar su correcto funcionamiento, se tomarán como referencia 3 elementos, uno de cada RPI: la iluminación de la cafetería, el horno de la pastelería y la calefacción del taller. Los primeros resultados obtenidos son los mostrados en la Figura 33.

Figura 33. Resultados experimentales de los elementos electrónicos en un escenario *concerned*.

Como se observa, en la hora 7 se activan la iluminación externa de la cafetería y la calefacción del taller, y en la hora 4 se activa el horno de la pastelería, coincidiendo con el escenario mostrado en la Figura 30, por lo que los primeros resultados son correctos. A continuación, se comprueban las horas en la que se tienen que desactivar dichos elementos, que son a las 13 para la iluminación de la cafetería y las 10 para el horno de la pastelería y la calefacción del taller, obteniendo los resultados mostrados en la Figura 34.

Figura 34. Resultados experimentales de los elementos electrónicos en un escenario *concerned*.

Se puede ver cómo los 3 elementos se han desactivado en las horas que les corresponden, por lo que se deduce que el escenario *concerned* ha funcionado correctamente. Se carga ahora un escenario *demanding*, con carga de trabajo y algún pico a lo largo del día. El escenario es el mostrado en la Figura 35.

1	Hora	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2																									
3	Cafeteria																								
4	Iluminacion	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	1	1	0	0
5	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
6	Horno	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1
7																									
8	Pasteleria																								
9	Iluminacion	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0
10	AC	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0
11	Riego auto	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0
12	Horno	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1
13																									
14	Taller																								
15	Iluminacion	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	1	1	1	1	0	0	0
16	AC	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
17	Alarma	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	0	0	0	0
18	Calefaccion	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1
19																									
20	Casa 1																								
21	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
22	Lavadodra	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
23	Riego	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
24																									
25	Dentista																								
26	Proyector	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	1	1	1	1	0	0	0	0	0
27	Iluminacion	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0
28	AC	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	0	0	0	0
29																									
30	Casa 2																								
31	Riego auto.	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
32	Calefaccion	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
33	AC	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	1	0	0
34	Lavadora	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
35																									
36	Casa 3																								
37	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
38	Lavaplatos	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
39	AC	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
40																									
41	Casa 4																								
42	Riego auto.	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
43	Lavaplatos	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
44	Horno	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0

Figura 35. Escenario *demanding* para validación del montaje.

La gráfica para verse de manera visual es la mostrada en la Figura 36. Para este caso, se estudiará la hora 19 al tener un pico de consumo muy alto y, por lo tanto, de mucho consumo. Los resultados obtenidos son los mostrados en la Figura 37.

Figura 36. Gráfica del escenario *demanding* simulado en el montaje.

Figura 37. Resultados experimentales de los elementos electrónicos en un escenario *demanding*.

Como se observa, todos los elementos se encuentran activos, como así indica la programación horaria. Se comprueba ahora la hora 22 para la iluminación, la hora 1 (del día siguiente) para el horno y la calefacción, en los que dichos elementos deben quedar inactivos.

Se puede comprobar también en la Figura 38 que los resultados responden a lo esperado.

Figura 38. Resultados experimentales de los elementos electrónicos en un escenario *demanding*.

Y, por último, se carga un escenario *busy*, mostrado en la Figura 39 en el que, como se ha mencionado anteriormente, presenta mucho volumen y es difícil de ajustar para el agregador.

1	Hora	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2																									
3	Cafeteria																								
4	Iluminacion	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
5	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
6	Horno	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1
7																									
8	Pasteleria																								
9	Iluminacion	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
10	AC	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
11	Riego auto	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
12	Horno	1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1
13																									
14	Taller																								
15	Iluminacion	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
16	AC	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
17	Alarma	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	0	0	0	0
18	Calefaccion	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1
19																									
20	Casa 1																								
21	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
22	Lavadodra	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
23	Riego	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
24																									
25	Dentista																								
26	Proyector	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
27	Iluminacion	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
28	AC	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	0	0	0	0
29																									
30	Casa 2																								
31	Riego auto.	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
32	Calefaccion	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
33	AC	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	1	0	0	0	0	0	0	0
34	Lavadora	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
35																									
36	Casa 3																								
37	Calefaccion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
38	Lavaplatos	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
39	AC	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
40																									
41	Casa 4																								
42	Riego auto.	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1
43	Lavaplatos	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
44	Horno	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

Figura 39. Escenario *busy* para validación del montaje.

La gráfica mostrada en la Figura 40 permite ver de manera visual dicho escenario, en el que se pude ver cómo hay gran cantidad de volumen desde las 7 horas hasta las 23.

Figura 40. Gráfica del escenario *busy* simulado en el montaje.

Al igual que en el caso anterior, se comprueba la hora 8 que es la hora de mayor consumo. Se obtienen los resultados mostrados en la Figura 41.

Figura 41. Resultados experimentales de los elementos electrónicos en un escenario *busy*.

Se comprueba ahora que se desactivan correctamente. Para ello, se comprueban en la hora 1, en la que todos los elementos deben permanecer desactivados, obteniendo los resultados mostrados en la Figura 42.

Figura 42. Resultados experimentales de los elementos electrónicos en un escenario *busy*.

Como se observa, se obtienen los resultados esperados ya que todos los elementos están desactivados, por lo que el montaje ha respondido a lo esperado tanto para

escenarios con más volumen y picos de consumo como para los que tienen menos volumen.

4- Conclusiones y trabajo futuro

4.1- Resumen y conclusiones

El proyecto “Eneff-Pilot” trata de demostrar que la sostenibilidad de las comunidades de consumidores de electricidad es más efectiva si se realiza de forma cooperativa que en enfoques individualistas. En la arquitectura de este proyecto se tienen 3 dominios fundamentales: el consumidor, la comunidad de propietarios en la que se encuentra el agregador de flexibilidad de la demanda y la comercializadora de la electricidad. Pues bien, en este trabajo se ha pretendido mostrar de manera visual el dominio del consumidor y el dominio del agregador con el fin de que sea más sencillo de entender la metodología de este proyecto.

Para mostrar el dominio del consumidor, se ha hecho un montaje mediante juegos de piezas Lego debido a la sencillez para la construcción, la variedad que había en el mercado y la sencillez para poder modificar e incluir los elementos electrónicos.

Dichos elementos electrónicos son controlados mediante 3 RPIs, las cuales reciben la programación horaria y son las encargadas de encender o apagar los aparatos cuando corresponde. Aunque un agregador manda órdenes directas a un controlador situado en el hogar y no directamente a los elementos electrónicos, se ha elegido este método ya que permite verse más sencillamente de manera visual y a efectos es lo mismo. Para establecer la programación se ha diseñado un software que permite cargar de manera automática o manual la programación por parte del administrador de la comunidad, el cual tratará de hacerlo de la manera más eficiente posible teniendo en cuenta el suministro de renovables.

4.2- Líneas futuras

Dada la motivación didáctica de este trabajo, se podría diseñar una APP de Android o iOS que mandara órdenes directas a la RPI de cada establecimiento, pudiéndose descargar y manejar de manera sencilla por cada consumidor.

Además, se podrían establecer tramos horarios de 5 minutos en lugar de horas enteras con el fin de que fueran más precisos, e incluir el consumo de cada aparato para controlar también el consumo por establecimiento.

El Lego también podría ser mejorado cubriendo al 100% su base y no dejando visible ninguna parte del cableado incluyendo, por ejemplo, un césped artificial en los alrededores. Se podrían incluir también elementos distintos tales como luces con sensor de movimiento o un indicador que marque en cada momento el consumo de potencia del hogar para poder reducirlo cuando pase ciertos límites.

Referencias

[1] "Stepper Motor 5V 4-Phase 5-Wire & ULN2003 Driver Board for Arduino" URL: http://eeshop.unl.edu/pdf/Stepper+Driver.pdf Accesed: 11/18/16

[2] Richardson, M., & Wallace, S. (2012). *Getting started with Raspberry PI*. " O'Reilly Media, Inc.".

[3] REIS, Inês FG, et al. "A multi-agent system approach to exploit demand-side flexibility in an energy community". *Utilities Policy*, 2020, vol. 67, p. 101114.

[4] KOIRALA, Binod Prasad, et al. Energetic communities for community energy: A review of key issues and trends shaping integrated community energy systems. *Renewable and Sustainable Energy Reviews*, 2016, vol. 56, p. 722-744.

[5] GOOD, Nicholas; MANCARELLA, Pierluigi. Flexibility in multi-energy communities with electrical and thermal storage: A stochastic, robust approach for multi-service demand response. IEEE Transactions on Smart Grid, 2017, vol. 10, no 1, p. 503-513.

[6] Rick Fredici, "Core Concepts and Leed Guide" Edición 2, p.19

[7] S. Kirbas et al., "On The Introduction of Automatic Program Repair in Bloomberg," in IEEE Software, vol. 38, no. 4, pp. 43-51, July-Aug. 2021, doi: 10.1109/MS.2021.3071086.

[8] Wayes Tushar, Lan Lan, Chathura Withanage, Hui En Karen Sng, Chau Yuen, Kristin L. Wood, Tapan Kumar Saha. "Exploiting design thinking to improve energy efficiency of buildings" Volume 197, 2020

[9] Joe Hanson, "LEGO model Smart home". URL: https://www.raspberrypi.com/news/lego-model-smart-home/ 2015 [Accesed: 27-Feb-2022].

[10] S. Jindarat and P. Wuttidittachotti, "Smart farm monitoring using RPI Pi and Arduino," 2015 International Conference on Computer, Communications, and Control Technology (I4CT), 2015, pp. 284-288, doi: 10.1109/I4CT.2015.7219582.

[11] E. Carrillo, V. Benitez, C. Mendoza and J. Pacheco, "IoT framework for smart buildings with cloud computing," 2015 IEEE First International Smart Cities Conference (ISC2), 2015, pp. 1-6, doi: 10.1109/ISC2.2015.7366197.

[12] P. Kumar and Umesh Chandra Pati, "Arduino and RPI Pi based smart communication and control of home appliance system," 2016 Online International Conference on Green Engineering and Technologies (IC-GET), 2016, pp. 1-6, doi: 10.1109/GET.2016.7916808.

[13] Lago Aguado, D., & Gómez Fuentes, R. (2016). Inteligencia ambiental en el Internet de las cosas.

[14] Red Eléctrica Española. RedOS la app del operador. https://www.ree.es/es/actividades/operacion-del-sistema-electrico/redos-app-operador-sistema

[15] J. Sanchez Soberanes, B. Vicente, A. Perez Pasten. Diseño de página web para la empresa SAEMI.

[16] Ángel Cobo y Patricia Gómez, “PHP y mysql. Tecnología para el desarrollo de aplicaciones web”, 2005.

[17] Enerta, Eficiencia Energética: “Blog: Agregador de demanda”, URL: https://www.enertra.es/agregador-de-demanda/ Accesed: 18-Marzo-2021

[18] Carlos Cruz, Esther Palomar and Ignacio Bravo and Manuel Aleixandre. “Behavioural patterns in aggregated demand response developments for communities targeting renewables”, 2021.

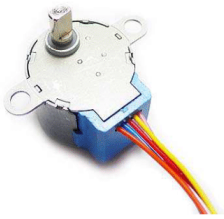
Anexos

Anexo I. Características del *28BYJ-48*

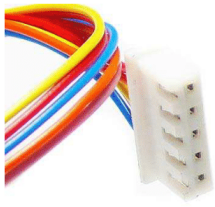
Kiatronics
electronic design and manufacture

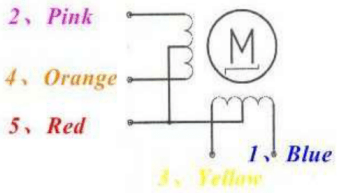
28BYJ-48 – 5V Stepper Motor

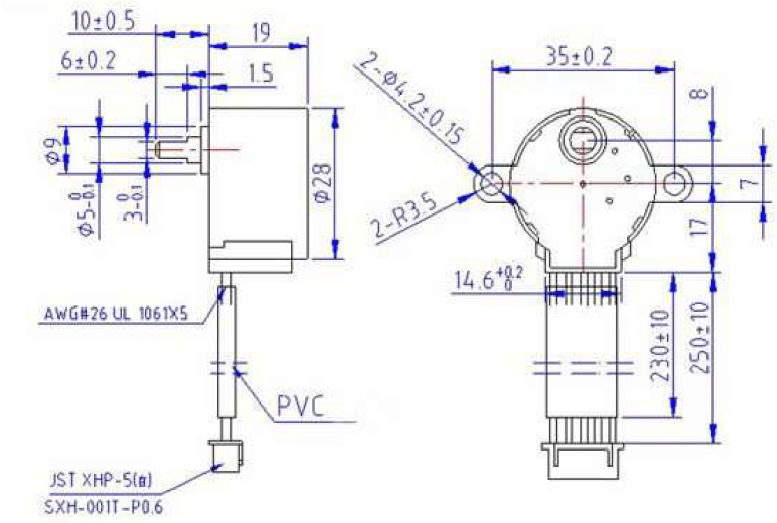
The 28BYJ-48 is a small stepper motor suitable for a large range of applications.



Rated voltage :	5VDC
Number of Phase	4
Speed Variation Ratio	1/64
Stride Angle	5.625°/64
Frequency	100Hz
DC resistance	50Ω±7%(25℃)
Idle In-traction Frequency	> 600Hz
Idle Out-traction Frequency	> 1000Hz
In-traction Torque	>34.3mN.m(120Hz)
Self-positioning Torque	>34.3mN.m
Friction torque	600-1200 gf.cm
Pull in torque	300 gf.cm
Insulated resistance	>10MΩ(500V)
Insulated electricity power	600VAC/1mA/1s
Insulation grade	A
Rise in Temperature	<40K(120Hz)
Noise	<35dB(120Hz,No load,10cm)
Model	28BYJ-48 – 5V







Anexo II. Consumo Diodos Led

5mm Clear Lens (I_F=20mA)	Forward Voltage / V	Dominant Wavelength / nm	Luminous Intensity / mcd	View Angle of Half Power / Degree
White	3.0~3.6	460	14250	15
Red	2.0~2.5	624 ± 3	5000 ± 20%	15 ± 3
Blue	3.5~4.0	467	5500	30
Yellow	2.0~2.5	590 ± 3	56000 ± 20%	15 ± 3
Green	3.6~4.0	520	8000	20
UV	3.5~4.0	405	250	30

Anexo III. Especificaciones RPI

Raspberry Pi 3 Model B+

2

Specifications

Processor: Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4 GHz

Memory: 1GB LPDDR2 SDRAM

Connectivity:
- 2.4 GHz and 5 GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- 4 × USB 2.0 ports

Access: Extended 40-pin GPIO header

Video & sound:
- 1 × full size HDMI
- MIPI DSI display port
- MIPI CSI camera port
- 4 pole stereo output and composite video port

Multimedia: H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics

SD card support: Micro SD format for loading operating system and data storage

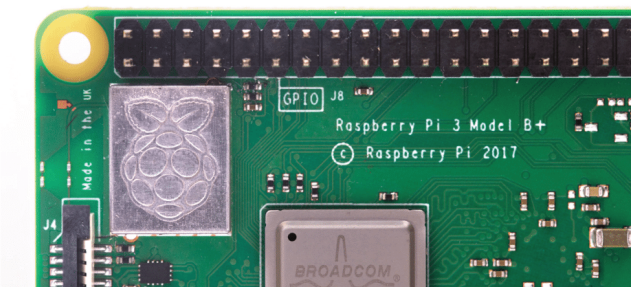
Input power:
- 5 V/2.5 A DC via micro USB connector
- 5 V DC via GPIO header
- Power over Ethernet (PoE)–enabled (requires separate PoE HAT)

Environment: Operating temperature, 0–50 °C

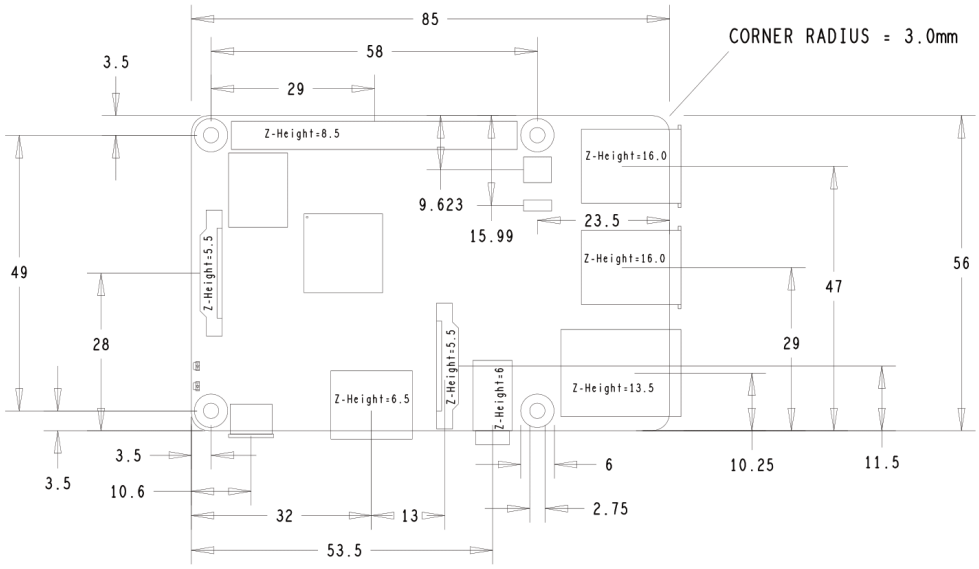
Compliance: For a full list of local and regional product approvals, please visit www.raspberrypi.org/products/raspberry-pi-3-model-b+

Production lifetime: The Raspberry Pi 3 Model B+ will remain in production until at least January 2023.

raspberrypi.org



Physical specifications



Warnings

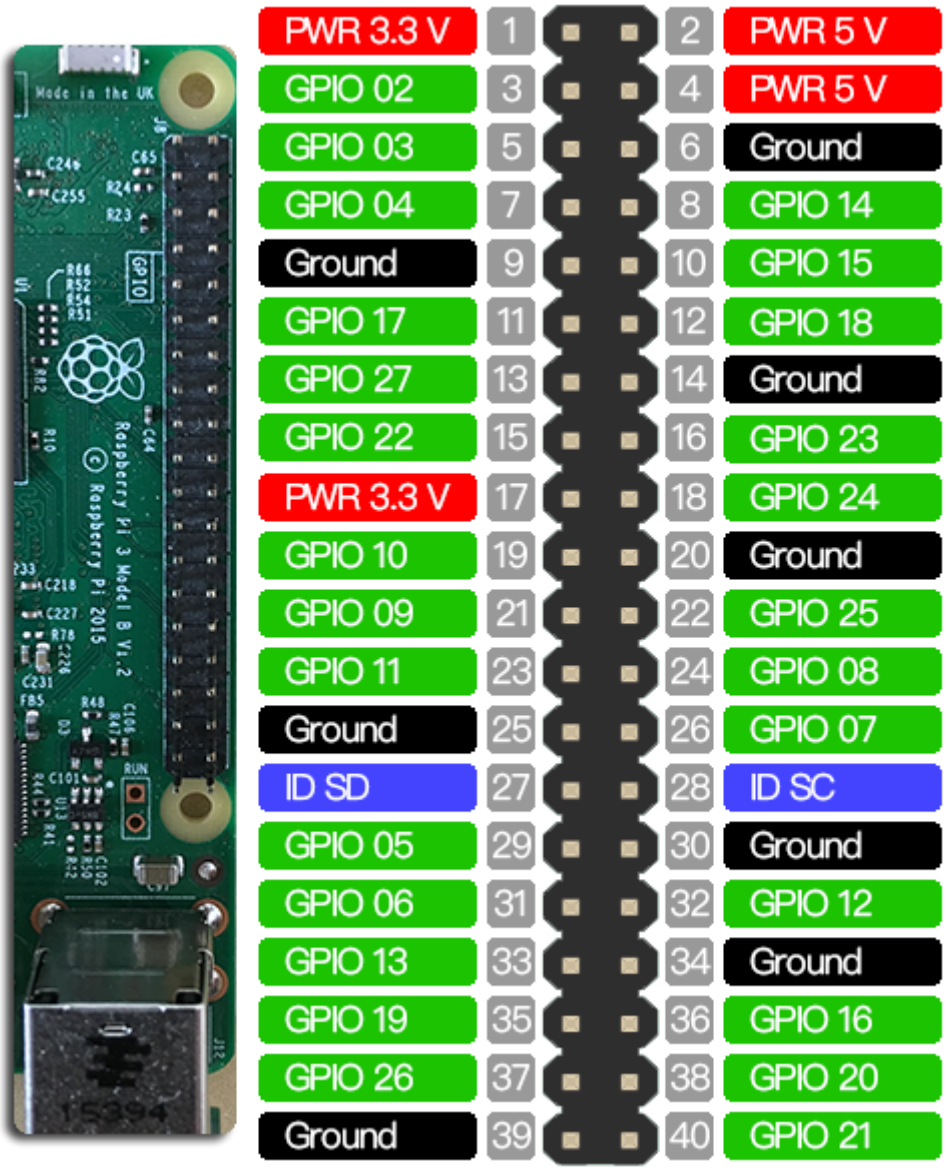
- This product should only be connected to an external power supply rated at 5 V/2.5 A DC. Any external power supply used with the Raspberry Pi 3 Model B+ shall comply with relevant regulations and standards applicable in the country of intended use.
- This product should be operated in a well-ventilated environment and, if used inside a case, the case should not be covered.
- Whilst in use, this product should be placed on a stable, flat, non-conductive surface and should not be contacted by conductive items.
- The connection of incompatible devices to the GPIO connection may affect compliance, result in damage to the unit, and invalidate the warranty.
- All peripherals used with this product should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met. These articles include but are not limited to keyboards, monitors, and mice when used in conjunction with the Raspberry Pi.
- The cables and connectors of all peripherals used with this product must have adequate insulation so that relevant safety requirements are met.

Safety instructions

To avoid malfunction of or damage to this product, please observe the following:

- Do not expose to water or moisture, or place on a conductive surface whilst in operation.
- Do not expose to heat from any source; the Raspberry Pi 3 Model B+ is designed for reliable operation at normal ambient temperatures.
- Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- Whilst it is powered, avoid handling the printed circuit board, or only handle it by the edges to minimise the risk of electrostatic discharge damage.

raspberrypi.org


PWR 3.3 V 1 2 PWR 5 V
GPIO 02 3 4 PWR 5 V
GPIO 03 5 6 Ground
GPIO 04 7 8 GPIO 14
Ground 9 10 GPIO 15
GPIO 17 11 12 GPIO 18
GPIO 27 13 14 Ground
GPIO 22 15 16 GPIO 23
PWR 3.3 V 17 18 GPIO 24
GPIO 10 19 20 Ground
GPIO 09 21 22 GPIO 25
GPIO 11 23 24 GPIO 08
Ground 25 26 GPIO 07
ID SD 27 28 ID SC
GPIO 05 29 30 Ground
GPIO 06 31 32 GPIO 12
GPIO 13 33 34 Ground
GPIO 19 35 36 GPIO 16
GPIO 26 37 38 GPIO 20
Ground 39 40 GPIO 21

Anexo IV. Código del fichero "*home.php*"

```
<?php
session_start();
if (!isset($_SESSION['luser']))
{
    sleep(2);
    header("Location:index.php");
}
else
{
    $now = time(); // Checking the time now when home page starts.

    if ($now > $_SESSION['expire'])
    {
        session_destroy();
        header("Location:index.php");
    }
    else { //Starting this else one [else1]
        $_SESSION['expire'] = time() + (10 * 60);
        ?>  <!--  HTML  -->

<?php include('database_home.php');?>

<!DOCTYPE HTML>
<html2 lang="es" >
<head>
       <title>Comunidad inteligente</title>
       <link rel="stylesheet" type="text/css" href="estilos.css">
       <link rel="icon" href="icono.png" type="image/gif">
       <?php include('STYLE.php');?>
       <meta http-equiv="Content-Type" content="text/html;charset=utf-8">

       <script src="jquery-3.3.1.min.js"></script>
       <script language="javascript" type="text/javascript">

              function DestruirSesion()
                     {
                         location.href = "logout.php";
                     }




              $(document).ready(function()
              {


                     function update(){
                            $.ajax({
```

```
                        url: 'ajax_db2.php',
                        data: "",
                        dataType: 'json',
                        cache: false,
                        success: true,
                        success: function(data)
                        {
                                if(data[0]==0)
                                {

$('#headingg3').css({"display":""});

$('#headingg4').css({"display":"none"});

$('#headingg5').css({"display":"none"});

$('#headingg6').css({"display":"none"});

$('#headingg7').css({"display":"none"});

$('#headingg8').css({"display":"none"});

$('#headingg9').css({"display":"none"});

$('#headingg10').css({"display":"none"});

$('#headingg11').css({"display":"none"});

$('#headingg12').css({"display":"none"});

$('#headingg13').css({"display":"none"});

$('#headingg14').css({"display":"none"});

$('#headingg15').css({"display":"none"});

$('#headingg16').css({"display":"none"});

$('#headingg17').css({"display":"none"});

$('#headingg18').css({"display":"none"});

$('#headingg19').css({"display":"none"});

$('#headingg20').css({"display":"none"});

$('#headingg21').css({"display":"none"});

$('#headingg22').css({"display":"none"});

$('#headingg23').css({"display":"none"});

$('#headingg24').css({"display":"none"});

$('#headingg25').css({"display":"none"});

$('#headingg26').css({"display":"none"});
```

```
                                        }

                                            . . .

                                                setInterval(update,1000);
            });


      </script>

      </head>

      <body onload="myfunction()"; style="background-color:#B7C5FE; font-
size: 12 px; "; >




<div style="width:70%; margin-left:auto;margin-right:auto; height:100%;">
  <style>
    .heading { color: #0A2AAD;  font:oblique bold 120%;}
  </style>
    <style>
    .heading2 { font: oblique bold 120% cursive;}
  </style>
  <style>
    .heading3 { color: green;  font:oblique bold 120%;}
  </style>

<b>

<div class="heading" style=" width:930px; margin-left:10%; font-
size:40px;">Comunidad de electricidad agregada
             
;  
<a href="/logout.php"><img src="apagado.png" style="width:40px"></a>
</div></b>
<b><div class="heading2" id="heading2"; style=" width:920px; margin-
left:auto; margin-right:auto; font-size:20px;">Pinche sobre un
establecimiento para planificar su programaci&oacuten horaria o cargue un
fichero</div></b>
<b><div class="heading2" id="heading3"; style=" display:none;width:70px;
margin-left:auto; margin-right:auto; font-
size:20px;">Cafeter&iacute;a</div></b>
<b><div class="heading2" id="heading4"; style=" display:none;width:70px;
margin-left:auto; margin-right:auto; font-
size:20px;">Pasteler&iacute;a</div></b>
<b><div class="heading2" id="heading5"; style=" display:none;width:70px;
margin-left:auto; margin-right:auto; font-size:20px;">Taller</div></b>
<b><div class="heading2" id="heading6"; style=" display:none;width:70px;
margin-left:auto; margin-right:auto; font-size:20px;">Casa 1</div></b>
<b><div class="heading2" id="heading7"; style=" display:none;width:70px;
margin-left:auto; margin-right:auto; font-size:20px;">Dentista</div></b>
<b><div class="heading2" id="heading8"; style=" display:none;width:70px;
margin-left:auto; margin-right:auto; font-size:20px;">Casa 2</div></b>
<b><div class="heading2" id="heading9"; style=" display:none;width:70px;
margin-left:auto; margin-right:auto; font-size:20px;">Casa 3</div></b>
```

```
<b><div class="heading2" id="heading10"; style="display:none;width:70px;
margin-left:auto; margin-right:auto; font-size:20px;">Casa 4</div></b>

<br>
</div>



<div style="width:900px; margin-left:auto;margin-right:auto;">

<img src="Casa.jpeg" style="margin-left:auto; margin-right:auto; width:100%"
>

<a href="cafeteria.php"><img src="Fondos.png" style="position:absolute;
width:220px; margin-left: -900px; height:23%;margin-top:320px; opacity:0"
onmouseout="cafeteria2();" onmouseover="cafeteria();"></a>
<a href="pasteleria.php"><img src="Fondos.png" style="position:absolute;
width:300px; margin-left: -670px; height:24%;margin-top:315px; opacity:0"
onmouseout="pasteleria2();" onmouseover="pasteleria();"></a>
<a href="taller.php"><img src="Fondos.png" style="position:absolute;
width:320px; margin-left: -340px; height:24%;margin-top:315px; opacity:0"
onmouseout="taller2();" onmouseover="taller();"></a>

<a href="casa1.php"><img src="Fondos.png" style="position:absolute;
width:220px; margin-left: -900px; height:17%;margin-top:205px; opacity:0"
onmouseout="casa12();" onmouseover="casa1();"></a>
<a href="dentista.php"><img src="Fondos.png" style="position:absolute;
width:300px; margin-left: -670px; height:17%;margin-top:200px; opacity:0"
onmouseout="dentista2();" onmouseover="dentista();"></a>
<a href="casa2.php"><img src="Fondos.png" style="position:absolute;
width:320px; margin-left: -340px; height:20%;margin-top:170px; opacity:0"
onmouseout="casa22();" onmouseover="casa2();"></a>

<a href="casa3.php"><img src="Fondos.png" style="position:absolute;
width:220px; margin-left: -900px; height:19%;margin-top:75px; opacity:0"
onmouseout="casa32();" onmouseover="casa3();"></a>
<a href="casa4.php"><img src="Fondos.png" style="position:absolute;
width:300px; margin-left: -670px; height:18%;margin-top:80px; opacity:0"
onmouseout="casa42();" onmouseover="casa4();"></a>
<img src="Fondos2.png" style="position:absolute; background-color:#B7C5FE;
width:300px; margin-left: -300px; height:9%;margin-top:0px; opacity:1" ">
<b><div class="headingg3" id="headingg3" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 0</div></b>
<b><div class="headingg3" id="headingg4" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 1</div></b>
<b><div class="headingg3" id="headingg5" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 2</div></b>
<b><div class="headingg3" id="headingg6" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 3</div></b>
<b><div class="headingg3" id="headingg7" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 4</div></b>
<b><div class="headingg3" id="headingg8" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 5</div></b>
```

```
<b><div class="headingg3" id="headingg9" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 6</div></b>
<b><div class="headingg3" id="headingg10" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 7</div></b>
<b><div class="headingg3" id="headingg11" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 8</div></b>
<b><div class="headingg3" id="headingg12" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 9</div></b>
<b><div class="headingg3" id="headingg13" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 10</div></b>
<b><div class="headingg3" id="headingg14" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 11</div></b>
<b><div class="headingg3" id="headingg15" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 12</div></b>
<b><div class="headingg3" id="headingg16" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 13</div></b>
<b><div class="headingg3" id="headingg17" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 14</div></b>
<b><div class="headingg3" id="headingg18" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 15</div></b>
<b><div class="headingg3" id="headingg19" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 16</div></b>
<b><div class="headingg3" id="headingg20" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 17</div></b>
<b><div class="headingg3" id="headingg21" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 18</div></b>
<b><div class="headingg3" id="headingg22" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 19</div></b>
<b><div class="headingg3" id="headingg23" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 20</div></b>
<b><div class="headingg3" id="headingg24" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 21</div></b>
<b><div class="headingg3" id="headingg25" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 22</div></b>
<b><div class="headingg3" id="headingg26" style="display:none; position:
absolute; width:300px; margin-left: 660px; margin-top:-480px; font-
size:30px;">Hora 23</div></b>


</div>
```

```
                    <div style="width:605px; margin-left:auto; margin-
right:auto;">
                            <table style="width:100%;">
                            <form method="post" action="/descargarCSV.php">
                                    <button class="btn10" type="submit"
style="width:160px; height:30px;"><b>Descargar
programaci&oacute;n</b></button>

                                </form>
                            


       <form enctype="multipart/form-data" action="upload.php" method="POST"
>
    <input type="hidden" name="MAX_FILE_SIZE" value="512000" />
    <input name="userfile" onchange="set_file()"; type="file" / >
    <b><input type="submit" id="set_files" class="btn10"
style="height:30px;font-weight:bold;" value="SET programaci&oacute;n" ></b>
    </form>
                                   

                            </table>
             </div>
```

Anexo V. Código del fichero “controlador.c”

```
#include <mysql.h>
#include <my_global.h>
#include <stdio.h>
#include <stdlib.h>
#include <bcm2835.h>
#include <wiringPi.h>
#include <string.h>
#include <math.h>

int main() {
MYSQL *conn, *con2, *con3;
 MYSQL_RES *res, *result, *result2;
 MYSQL_ROW row, row2;
 int motor1=0, motor2=0, horas=0;

 char query[1500]={0}, *comando;

 char *server = "localhost";char *server2 = "192.168.1.12";char *server3 =
"192.168.1.14";
  char *user = "phpmyadmin";
  char *password = "phpseratelpiphp";
  char *database = "db_mux";
  con2 = mysql_init(NULL); conn = mysql_init(NULL); con3 = mysql_init(NULL);
  if (mysql_real_connect(con2, "localhost", "phpmyadmin", "phpseratelpiphp",
"db_mux", 0, NULL, 0) == NULL){mysql_error(con2);}
  if (mysql_real_connect(conn, "192.168.1.12", "phpmyadmin",
"phpseratelpiphp", "db_mux", 0, NULL, 0) == NULL){mysql_error(conn);}//RPI2
  if (mysql_real_connect(con3, "192.168.1.14", "phpmyadmin",
"phpseratelpiphp", "db_mux", 0, NULL, 0) == NULL){mysql_error(con3);}//RPI3

    bcm2835_init();

    bcm2835_gpio_fsel(21, BCM2835_GPIO_FSEL_OUTP);   //Calefaccion piso 1
izquierda RPI 1
    bcm2835_gpio_fsel(2, BCM2835_GPIO_FSEL_OUTP);   //Horno casa abajo a la
izquierda RPI 1
    bcm2835_gpio_fsel(3, BCM2835_GPIO_FSEL_OUTP);  //Calefacción casa abajo a
la izquierda RPI 1

    bcm2835_gpio_fsel(26, BCM2835_GPIO_FSEL_OUTP);   //Iluminación externa
casa abajo a la izquierda RPI 1

    bcm2835_gpio_fsel(5, BCM2835_GPIO_FSEL_OUTP);    //Sistema riego piso 1
izquierda RPI1

    bcm2835_gpio_fsel(17, BCM2835_GPIO_FSEL_OUTP);   //Calefaccion piso 2
izquierda RPI 1
    bcm2835_gpio_fsel(27, BCM2835_GPIO_FSEL_OUTP);   //Lavaplatos piso 2
izquierda RPI 1



while(1){
int num_motor, pin;
```

```
if (mysql_query(con2, "SELECT hora_ FROM Hora"))
{mysql_error(con2);}
result2 = mysql_store_result(con2);
if (result2 == NULL)
{mysql_error(con2);}
row2 = mysql_fetch_row(result2);
horas=atoi(row2[0]);


delay(2000); //retardo de hora



      if(atoi(row2[0])==0) //hora 0
      {
      if (mysql_query(con2, "SELECT hora0_cafeteria_cal,
hora0_cafeteria_horno, hora0_cafeteria_iluminacion, hora0_casa1_cal,
hora0_casa1_lavadora, hora0_casa1_riego, hora0_casa3_cal,
hora0_casa3_lavaplatos, hora0_casa3_AC FROM Cafeteria_calefaccion,
Cafeteria_horno, Cafeteria_ledfachada, Casa1_calefaccion, Casa1_lavadora,
Casa1_riego, Casa3_calefaccion, Casa3_lavaplatos, Casa3_ventilador"))
      {mysql_error(con2);}
      result2 = mysql_store_result(con2);
      if (result2 == NULL)
      {mysql_error(con2);}
      row2 = mysql_fetch_row(result2);

      if(atoi(row2[0])==1)            //calefaccion caf
            bcm2835_gpio_write(3, 1);
      else
            bcm2835_gpio_write(3, 0);

      if(atoi(row2[1])==1)      //horno caf
            bcm2835_gpio_write(2, 1);
      else
            bcm2835_gpio_write(2, 0);

      if(atoi(row2[2])==1)      //iluminacion caf
            bcm2835_gpio_write(26, 1);
      else
            bcm2835_gpio_write(26, 0);

      if(atoi(row2[3])==1)            // calefaccion casa1
            bcm2835_gpio_write(21, 1);
      else
            bcm2835_gpio_write(21, 0);

      if(atoi(row2[4])==1)            //lavadora casa1
            {sprintf(query, "UPDATE Motores_rpi1 SET motor_lavadora_rpi1
='1'");mysql_query(con2, query);}
      else
            {sprintf(query, "UPDATE Motores_rpi1 SET motor_lavadora_rpi1
='0'");mysql_query(con2, query);}

      if(atoi(row2[5])==1)            //riego casa 1
            bcm2835_gpio_write(5, 1);
      else
            bcm2835_gpio_write(5, 0);
```

```
	if(atoi(row2[6])==1)	//calefaccion casa3
		bcm2835_gpio_write(17, 1);
	else
		bcm2835_gpio_write(17, 0);

	if(atoi(row2[7])==1)	//lavaplatos casa3
		bcm2835_gpio_write(27, 1);
	else
		bcm2835_gpio_write(27, 0);

	if(atoi(row2[8])==1)	//AC casa3
		{sprintf(query, "UPDATE Motores_rpi1 SET motor_AC_rpi1
='1'");mysql_query(con2, query);}
	else
		{sprintf(query, "UPDATE Motores_rpi1 SET motor_AC_rpi1
='0'");mysql_query(con2, query);}



}

...

delay(50);
horas++;
 if(horas>23)
 horas=0;
sprintf(query, "UPDATE Hora SET  hora_ = %d", horas);mysql_query(con2,
query);mysql_query(conn, query);mysql_query(con3, query);
 delay(50);


	}

}
```

Anexo VI. Código del fichero “motores.c”

```
#include <mysql.h>
#include <my_global.h>
#include <stdio.h>
#include <stdlib.h>
#include <bcm2835.h>
#include <wiringPi.h>
#include <string.h>
#include <math.h>
void motor1 (int dir, int pasos, int motor_0, int motor_1)
{
	int pin, _step=0, i=0;
	   for(i=0; i<pasos; i++)
	   {


	    switch(_step){
	    case 0:
	          if(motor_0==1){

	          bcm2835_gpio_write(19, LOW);
	          bcm2835_gpio_write(13, LOW);
	          bcm2835_gpio_write(6, LOW);
	          }

	          if(motor_1==1){

	          bcm2835_gpio_write(19, LOW);
	          bcm2835_gpio_write(13, LOW);
	          bcm2835_gpio_write(4, LOW);
	          }

	    break;
	    case 1:
	    if(motor_0==1){
	    bcm2835_gpio_write(19, LOW);
	    bcm2835_gpio_write(13, LOW);
	    bcm2835_gpio_write(6, HIGH);
	    }


	    if(motor_1==1){
	    bcm2835_gpio_write(19, LOW);
	    bcm2835_gpio_write(13, LOW);
	    bcm2835_gpio_write(4, HIGH);
	    }

	    break;
	    case 2:
	    if(motor_0==1){
	    bcm2835_gpio_write(19, LOW);
	    bcm2835_gpio_write(13, LOW);
	    bcm2835_gpio_write(6, HIGH);
	    }
```

```
if(motor_1==1){
bcm2835_gpio_write(19, LOW);
bcm2835_gpio_write(13, LOW);
bcm2835_gpio_write(4, HIGH);
}

break;
case 3:
if(motor_0==1){
bcm2835_gpio_write(19, LOW);
bcm2835_gpio_write(13, HIGH);
bcm2835_gpio_write(6, HIGH);
}


if(motor_1==1){
bcm2835_gpio_write(19, LOW);
bcm2835_gpio_write(13, HIGH);
bcm2835_gpio_write(4, HIGH);
}


break;
case 4:
if(motor_0==1){
bcm2835_gpio_write(19, LOW);
bcm2835_gpio_write(13, HIGH);
bcm2835_gpio_write(6, LOW);
}


if(motor_1==1){
bcm2835_gpio_write(19, LOW);
bcm2835_gpio_write(13, HIGH);
bcm2835_gpio_write(4, LOW);
}


 break;
 case 5:
 if(motor_0==1){
 bcm2835_gpio_write(19, HIGH);
 bcm2835_gpio_write(13, HIGH);
 bcm2835_gpio_write(6, LOW);
 }


 if(motor_1==1){
 bcm2835_gpio_write(19, HIGH);
 bcm2835_gpio_write(13, HIGH);
 bcm2835_gpio_write(4, LOW);
 }


 break;
 case 6:
 if(motor_0==1){
 bcm2835_gpio_write(19, HIGH);
 bcm2835_gpio_write(13, LOW);
```

```
bcm2835_gpio_write(6, LOW);
}


if(motor_1==1){
bcm2835_gpio_write(19, HIGH);
bcm2835_gpio_write(13, LOW);
bcm2835_gpio_write(4, LOW);
}


break;
case 7:
if(motor_0==1){
bcm2835_gpio_write(19, HIGH);
bcm2835_gpio_write(13, LOW);
bcm2835_gpio_write(6, LOW);
}


if(motor_1==1){
bcm2835_gpio_write(19, HIGH);
bcm2835_gpio_write(13, LOW);
bcm2835_gpio_write(4, LOW);
}


break;
default:
       if(motor_0==1){
       bcm2835_gpio_write(19, LOW);
       bcm2835_gpio_write(13, LOW);
       bcm2835_gpio_write(6, LOW);
       }

       if(motor_1==1){
       bcm2835_gpio_write(19, LOW);
       bcm2835_gpio_write(13, LOW);
       bcm2835_gpio_write(4, LOW);
       }


break;
}
if(dir==0){
_step++;
}else{
_step--;
}
if(_step>7){
_step=0;
}
if(_step<0){
_step=7;
}
   delay(1);
}
```

```
}


int main() {
       MYSQL *conn, *con2, *con3;
        MYSQL_RES *res, *result, *result2;
        MYSQL_ROW row, row2;
        int count_horas=0, horas=0;

        char query[1500]={0}, *comando;

        char *server = "localhost";
         char *user = "phpmyadmin";
         char *password = "phpseratelpiphp";
         char *database = "db_mux";
         con3 = mysql_init(NULL);
         if (mysql_real_connect(con3, "localhost", "phpmyadmin",
"phpseratelpiphp", "db_mux", 0, NULL, 0) == NULL){mysql_error(con3);}

           bcm2835_init();
           bcm2835_gpio_fsel(19, BCM2835_GPIO_FSEL_OUTP);   //IN 1 motor
lavadora piso 1 izquierda RPI 1
           bcm2835_gpio_fsel(13, BCM2835_GPIO_FSEL_OUTP);   //IN 2 motor
lavadora piso 1 izquierda RPI 1
           bcm2835_gpio_fsel(6, BCM2835_GPIO_FSEL_OUTP);    //IN 3 motor
lavadora piso 1 izquierda RPI 1

           bcm2835_gpio_fsel(19, BCM2835_GPIO_FSEL_OUTP);   //IN 1 motor piso
2 izquierda ventilador RPI1
           bcm2835_gpio_fsel(13, BCM2835_GPIO_FSEL_OUTP);   //IN 2 motor piso
2 izquierda ventilador RPI1
           bcm2835_gpio_fsel(4, BCM2835_GPIO_FSEL_OUTP);    //IN 3 motor piso
2 izquierda ventilador RPI1



       while(1){
       int num_motor, pin, dir=0;
       int  _step=0, i=0;



             if (mysql_query(con3, "SELECT motor_lavadora_rpi1, motor_AC_rpi1
FROM Motores_rpi1"))
             {mysql_error(con3);}
             result2 = mysql_store_result(con3);
             if (result2 == NULL)
             {mysql_error(con3);}
             row2 = mysql_fetch_row(result2);

             if((atoi(row2[0])==1)||(atoi(row2[1])==1))
                    motor1(1, 500, atoi(row2[0]), atoi(row2[1]));

             delay(100);}
}
```

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR