Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Diseño e Implementación de un Sistema Domótico basado en Radioenlaces y Gestión Web Remota

Autor: Francisco Manuel Toro Cárdenas

Tutor: Juan Antonio Sánchez Segura

Dpto. Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2013

Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Diseño e Implementación de un Sistema Domótico basado en Radioenlaces y Gestión Web Remota

Autor:
Francisco Manuel Toro Cárdenas

Tutor:
Juan Antonio Sánchez Segura
Profesor colaborador

Dpto. de Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2019

Proyecto Fin de Carrera: Diseño e Implementación de un Sistema Domótico basado en Radioenlaces y Gestión Web Remota

Autor: Francisco Manuel Toro Cárdenas

Tutor: Juan Antonio Sánchez Segura

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

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2019

El Secretario del Tribunal

Agradecimientos

Tengo que reconocer que este camino habría sido imposible sin el apoyo de mi familia. Ellos creyeron en mí incluso cuando yo no lo hacía, tendiéndome su mano, y ayudándome en todo lo posible, tanto económica como emocionalmente. Por este motivo quiero agradecer a todos mis familiares y profesores haberme hecho un firme seguidor de la cultura del esfuerzo, que buscó una enseñanza más que un aprobado.

Muchas gracias a todos.

Francisco Manuel Toro Cárdenas

Sevilla, 2019

Resumen

Se partía de un montaje electrónico en Arduino, el cual, alojaba una página web encargada de activar un relé que alimentaba un mando a distancia de 433MHz. A su vez, en el lado receptor, se encontraba un receptor que activaba un relé alimentando así una bomba de agua. Debido a las carencias de seguridad que ofrecía la página web en arduino, se optó por cambiar dicho dispositivo por una Raspberry pi 3 Model B+, ya que esta puede emplear los certificados SSL para garantizar que lascomunicaciones vía web con el servidor sean totalmente seguras.

Se ha sustituido el mando y el receptor de 433MHz (comunicación simplex), que ofrecían unas opciones muy cerradas (encendido/apagado) por 2 pares de receptor-emisor de 433MHz que permiten una comunicación half-dúplex. Mediante la codificación de la transmisión, se ha conseguido controlar hasta un total de 16 equipos, destinando una parte a la identificación de dichos equipos y otra parte a la emisión de información relacionada con la ejecución deseada.

En líneas generales, se ha empleado la Raspberry Pi para albergar la página web. Esta se comunica mediante los módulos de 433MHz con el Arduino Nano situado en el lado receptor, el cual gestiona la activación de los equipos conectados a este.

Abstract

There was an electronic arduino assembly, which hosted a web page responsible for activating a relay that powered a 433MHz remote control. At the same time, on the receiving side, there was a receiver that activated a relay, thus feeding a water pump. Due to the lack of security offered by the web page in arduino, it was decided to change this device for a Raspberry Pi 3 Model B +, because it can use the SSL certificates to guarantee that the web communications with the server are totally secure.

The 433MHz remote control and receiver has been replaced (simplex communication), which offered few options (on / off) for 2 433MHz receiver-emitter pairs that allow a half-duplex communication. By coding the transmission, it has been possible to control up to a total of 16 equipment, assigning a part to the equipment identification and another part to the issuance of information related to the desired execution.

In general, it has been used the Raspberry pi to host the website. It communicates through the 433MHz modules with the Arduino Nano located on the receiver side, which manages the activation of the equipment connected to it.

Índice

Índice de Tablas

Índice de Figuras

Notación

SCE	Sistemas de Cableado Estructurado
TIC	Tecnología de informática y comunicaciones
GSM	Global System for Mobile communications
IR	Infrarrojos
RF	Radiofrecuencia
TX	Transmisión/transmitir
BW	Ancho de banda
MSE	Minimum Square Error
ASK	Amplitude Shift Keying
~	Aproximadamente
EDR	Enhanced Data Rate
HS	High Speed
IEEE	Institute of Electrical and Electronics Engineers
ISM	Industrial, Scientific and Medical
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
Li-Fi	Light-Fidelity
Wi-Fi	Wireless-Fidelity
SSL	Secure Socket Layer
IDE	Integrated Development Environment
SMD	Surface-mount Device
PCB	Printed Circuit Board
SD	Secure Digital
SSH	Secure Shell
IP	Internet Protocol
HTTP	Hypertext Transfer Protocol
PHP	Hypertext Preprocessor
HTML	HyperText Markup Language
NAT	Network Address Translation
DNS	Domain Name System
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
USB	Universal Serial Bus
RAM	Random Access Memory
HDMI	High-Definition Multimedia Interface
GPIO	General Purpose Input/Output
GPU	Unidad de procesamiento gráfico
SRAM	Statico Random Acces Memory
EEPROM	Electrically Erasable Programmable Read-Only Memory

1 INTRODUCCIÓN

Yo creo bastante en la suerte. Y he constatado que,
cuanto más duro trabajo, más suerte tengo.

Thomas Jefferson, 1743-1826

Se conoce como domótica a aquellos sistemas que permiten automatizar edificaciones de cualquier índole, cubriendo necesidades de energía, seguridad, comunicación y bienestar. Los dispositivos de la red pueden estar interconectados mediante redes cableadas o inalámbricas. El control de los sistemas actualmente goza de cierta ubicuidad, es decir, se pueden gestionar desde el interior del edificio, pero también telemáticamente. Domótica procede de la unión de *domus* (casa en latín) y autónomo ("que se gobierna a sí mismo" del latín).

Hoy en día la domótica está presente en gran cantidad de hogares cubriendo un gran abanico de posibilidades, desde sistemas sencillos hasta casas completamente automatizadas. Sin embargo, esta tecnología comenzó de una forma mucho más simple.

La domótica tiene sus orígenes a principios de los '70, época en la que aparecieron las primeras pruebas en viviendas. No obstante, no fue hasta los '80 cuando se empezaron a comercializar los primeros sistemas que conseguían integrar lo eléctrico, que manejaba todos los dispositivos del hogar, con lo electrónico.

La evolución de la informática implicó su expansión en los países de vanguardia como Estados Unidos, Alemania y Japón.

Con el ingente crecimiento de la informática en el hogar, llegaron los primeros edificios con Sistemas de Cableado Estructurado (SCE) que es de gran ayuda para formar redes de equipos.

El primer programa que utilizó la domótica fue el Save. Creado en Estados Unidos en 1984 con el fin de lograr una gran eficiencia y un bajo coste energético en los sistemas de control de los edificios domotizados. Para ello, dichas construcciones estaban dominadas por el sistema X-10, un protocolo de comunicaciones que podía ejecutarse desde un control remoto. Dicho sistema sigue siendo el más empleado desde que en 1976 Pico Electronics lo diseñase, debido a que al transmitir datos por las líneas de baja tensión, la relación costo-beneficio es la mejor opción posible.

En los últimos 30 años la domótica ha crecido sin cesar a gran escala. Esto tiene su explicación en la ventaja que supuso el desarrollo e implantación de las redes informáticas que posibilitaban la comunicación, ya sea de forma cableada o inalámbrica.

La llegada de la era TIC implica el estudio de las instalaciones domóticas en casa desde un punto de vista más realista. Actualmente, la domótica es un servicio muy consolidado. Sistemas de desarrollo 2.0 como el ZigBee permiten diseñar protocolos inalámbricos para la comunicación doméstica.

1.1. Tipos de instalaciones domóticas

Existe una amplia variedad de sistemas y protocolos en el mercado orientados a la domótica. Cada uno de ellos es diferente, aunque en general, los consumidores suelen tener unos requisitos comunes: que el sistema pueda aguantar el mayor número de equipos conectados y que sean interoperables entre sí. Además de estos dos factores, también hay otros aspectos bastante importantes a tener en cuenta, como el consumo, el ancho de

bando o el precio.

1.1.1 Sistemas inalámbricos

En las comunicaciones inalámbricas, las señales son transmitidas a través del aire sin el uso de ningún cable u otro material conductor. Hay bastantes tipos de comunicaciones inalámbricas, entre las que destacamos: IR (infrarrojos), a través de los canales destinados a la comunicación móvil, radiofrecuencia (RF), Wi-Fi, Bluetooth, Li-Fi, ZigBee, Zwave, láser y ultrasónicos.

Infrarrojos

En esta tecnología la información es transmitida por los equipos o sistemas a través de radiación infrarroja. En el espectro electromagnético, las ondas infrarrojas se encuentran entre las microondas y la luz visible. Son usadas para sistemas de seguridad, mandos de control remoto y comunicaciones de corto alcance.

Mediante la emisión de una señal modulada, el receptor puede leer y decodificar la señal obteniendo una serie de bits que corresponden al flujo binario modulado en el transmisor.

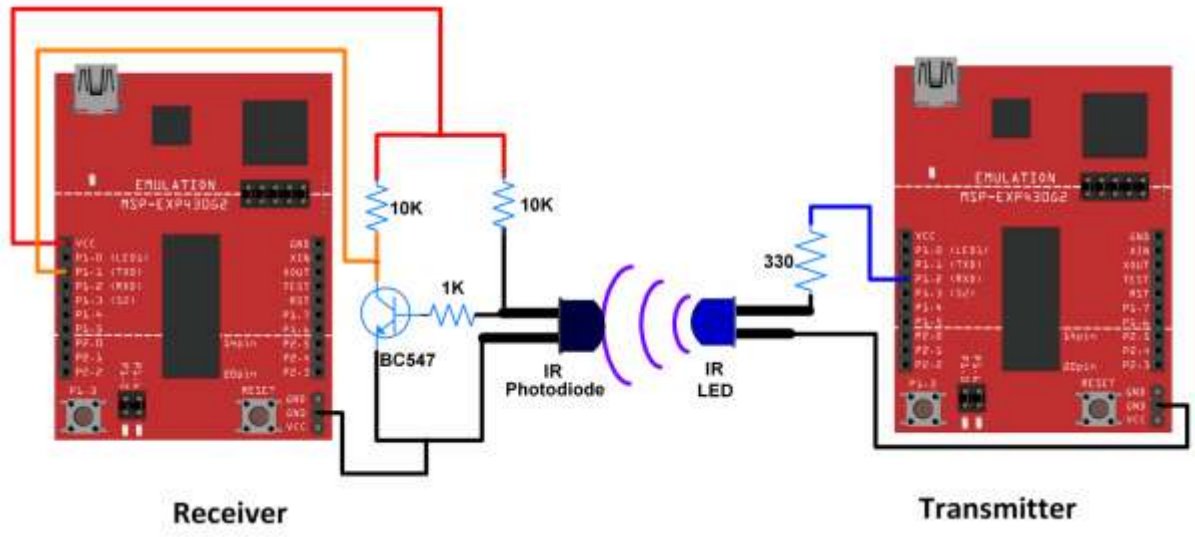

Figura 1.1 Ejemplo de comunicación a través de infrarrojos.

Los sistemas infrarrojos son usados en general para un rango corto y medio. Algunos de ellos operan en visión directa; esto significa que la comunicación no funcionará si hay un obstáculo entre emisor y receptor. Otros sistemas operan en modo difuso, es decir, operan aunque el emisor y receptor no estén en visión directa. Un ejemplo de esto último es el control remoto de la televisión, este no tiene que estar apuntando directamente al receptor, pero si tiene que estar en la misma habitación.

Una de las desventajas que presenta esta tecnología frente a otras como RF o Wi-Fi, es que los infrarrojos no pueden transmitirse a través de los muros o edificaciones en general, por lo que imposibilita la comunicación entre distintas habitaciones de un mismo edificio. Quizá esto podría verse como una desventaja, pero esto ofrece mucha más privacidad.

Teléfono móvil

Puede ser usado para controlar multitud de equipos del hogar. Funciona como un control remoto, aplicando tecnología Bluetooth, Wi-Fi ó GSM. En este último modo es posible controlar el sistema domótico desde casi cualquier parte del mundo, ya que hace uso de la extensa cobertura que proporciona la red móvil.

GSM fue originariamente diseñado para recibir y realizar llamadas y mensajes. Para domótica esto es más que suficiente para ejecutar bastantes de las tareas requeridas.

La comunicación mediante GSM es altamente recomendable para lugares en los que no se cuenta con ninguna instalación previa y se requiere conectividad mundial y a bajo coste. Un ejemplo perfecto de esta necesidad, seria la toma de valores de un sensor que tenemos en una explotación agrícula.

Figura 1.2 Módulo GSM para microcontrolador.

Wi-Fi

Este protocolo de comunicaciones inalámbricas está ampliamente extendido en domótica debido a la alta disponibilidad de este en el mercado de los dispositivos inteligentes. Permite controlar dispositivos a través de un Smartphone, ordenador personal u tablet mediante un sistema de comunicaciones bastante robusto. La mayoría de los dispositivos Wi-Fi usan la frecuencia de los 2.4GHz e implementan un multiplexado por división en frecuencia.

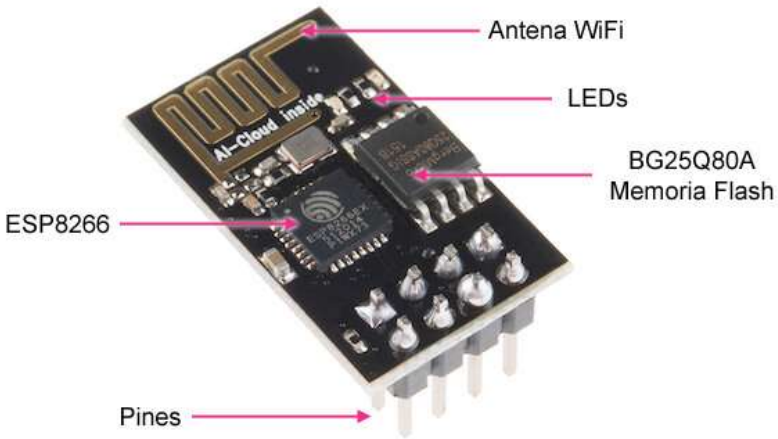

Figura 1.3 Módulo Wi-Fi para un microcontrolador.

La red Wi-Fi puede estar compuesta por multitud de equipos como: puntos de acceso, smartphones, módulos Wi-Fi para microcontroladores, repetidores, enrutadores…

Bluetooth

Al igual que el protocolo Wi-Fi es fácilmente accesible, ya que existe en multitud de equipos electrónicos, entre ellos el teléfono móvil.

Bluetooth es un protocolo específico cuyo destino es la aplicación en sistemas de comunicación de bajo consumo, con un coste tecnológico bajo y un rango de operación limitado.

Las comunicaciones suceden mediante radiofrecuencia, por lo que se asegura su operatividad entre habitaciones separadas o en general lugares que no tienen visión directa, siempre y cuando estén dentro del rango de alcance máximo. Existen varias clasificaciones, en función a su potencia de transmisión.

Tabla 1.1 Clasificación de Bluetooth respecto a su potencia tx.

Clase	Potencia máxima permitida (mW)	Potencia máxima Permitida (dBm)	Alcance (m)
1	100	20	~100
2	2,5	4	~5-10
3	1	0	~1
4	0,5	0	~0,5

Con respecto a la clasificación de Bluetooth en base a la capacidad del canal podemos diferenciarlos atendiendo a la siguiente tabla.

Tabla 1.2 Clasificación de Bluetooth respecto a su BW.

Versión	Ancho de banda (Mbps)
1.2	1
2.0+EDR	3
3.0+HS	24
4.0	32
5	50

Radiofrecuencia

Se puede controlar las aplicaciones domóticas mediante un par de módulos RF (transmisor y receptor). El módulo RF más usado es el de 433MHz, el cual se caracteriza por una comunicación simplex con canal único y unidireccional. Aunque no es necesario operarlos con procesadores como Arduino o Raspberry Pi, estos aportan mayor flexibilidad.

Los modelos más empleados de esta frecuencia son el FS1000A para el emisor y el XY-MK-5V para el receptor. El principal motivo de su popularidad es el bajo precio de estas unidades.

La frecuencia más usada es de 433MHz y aunque existen otros módulos que operan a 315MHz su uso está bastante menos extendido. Ambos modelos tienen la ventaja de operar en bandas libres, por lo que el uso es totalmente gratuito.

Ejemplos de uso típicos para este módulo es la obtención inalámbrica de datos como temperatura, presión, humedad y control de aplicaciones de domótica como: activación remota de dispositivos: iluminación, relés, alarmas… También se emplean en la monitorización de robots.

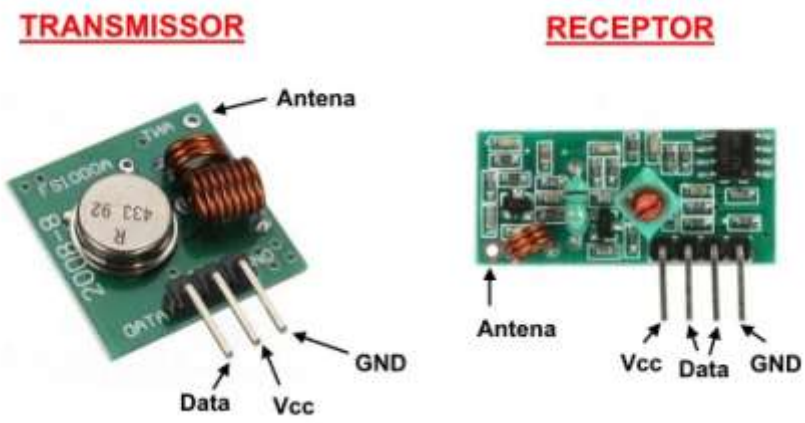

Figura 1.4 Transmisor y receptor.

ZigBee

Toma como base el nivel físico y el control de acceso al medio que están descritos en el estándar 802.15.4 del IEEE.

ZigBee emplea la banda ISM para el uso industrial, científico y médico. En Europa se sitúa en los 868MHz, mientras que en Estados unidos está en los 915MHz y en 2,4GHz para todo el mundo. Debido a que esta última banda es de uso libre, la mayoría de empresas tienden a usar Zigbee en esta banda para el uso y desarrollo de dispositivos. Funciona a través de 16 canales con un ancho de banda por canal de 5MHz. El protocolo que emplea para la transmisión es CSMA/CA para dotar de robustez y evitar las colisiones en las comunicaciones.

Esta tecnología se caracteriza por su bajo coste y por su reducido consumo energético, lo cual lo hace ideal para aplicaciones domóticas. Usa protocolos para la comunicación entre nodos que posibilitan la activación de estos cuando son usados y su posición a reposo cuando no, ahorrando gran cantidad de energía.

Las redes ZigBee pueden tener varias topologías: estrella, árbol y malla. La más usada es la topología en forma de malla, ya que permite que todos los nodos de la red estén interconectados con más de un equipo, salvando así la inoperatividad en caso de caída de un enlace. Para coordinar todas estas comunicaciones, debe haber un nodo que dirija todos los pasos de mensaje.

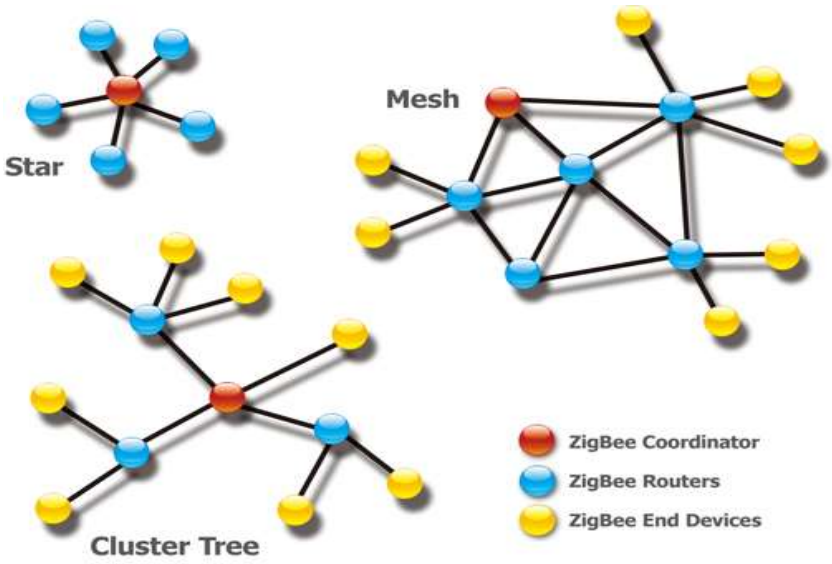

Figura 1.5 Topologías en una red ZigBee.

Dentro de la red ZigBee podemos clasificar distintos tipos de dispositivos:

- El coordinador ZigBee es el más importante e imprescindible, ya que se encarga de gestionar, sincronizar y encaminar todas las rutas necesarias para establecer las comunicaciones. Su existencia es obligatoria.
- El router ZigBee tiene la misión de conectar los nodos usuario.
- El dispositivo final o de usuario ZigBee solo recibe información y se comunica a través de su nodo superior.

Z-Wave

Es un protocolo empleado para domótica doméstica y de negocios. Su principal ventaja es que usa señales radio de baja potencia que permiten la comunicación entre los distintos equipos deseados sin que existan interferencias con los demás equipos existentes.

Aunque coexisten más protocolos en el mercado, Z-Wave presentan dos grandes ventajas: al operar en los 900MHz, y no en los 2,4GHz como Wi-Fi, presenta un mayor rendimiento, ya que las frecuencias más altas presentan más interferencias, y además, la penetración a través de objetos (muros, puertas, techos…) se mejora ya que la longitud de onda es mayor.

Z-Wave está desarrollado para comunicaciones robustas con una baja tasa de datos (hasta 100Kbps.

Normalmente las comunicaciones tienen un alcance de 30m, siendo posible hasta 4 saltos entre los nodos de paso.

Li-Fi

Es un sistema de transmisión inalámbrico bidireccional que se caracteriza por tener una mayor tasa de transmisión que Wi-Fi (en un futuro podrían alcanzarse los 15Gbps) empleando frecuencias de entre 400 y 800 THz (espectro visible).

El objetivo es poder establecer comunicaciones inalámbricas a la vez que se puede iluminar una

estancia con el empleo de elementos muy sencillos como son las bombillas led.

Los elementos principales de una comunicación por Li-Fi son: el modulador, el cual se encarga de apagar y encender el led de tal forma que sea imperceptible para el ojo humano y a la vez se estén transmitiendo ceros y unos que componen el mensaje, y el fotodiodo que es responsable de recibir las señales luminosas transmitidas y pasarlas al dominio eléctrico.

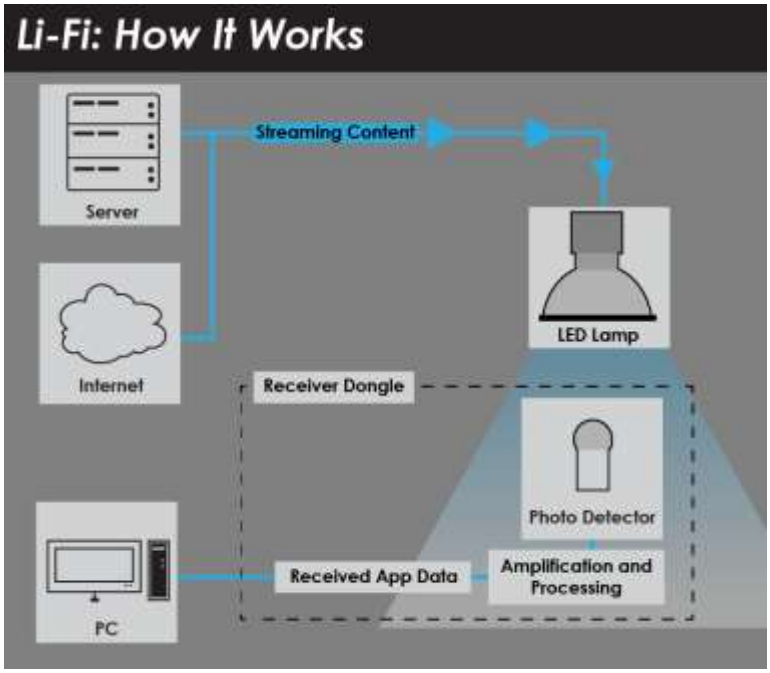

Figura 1.6 Funcionamiento de Li-Fi.

Li-Fi presenta la ventaja de no saturar el espectro actualmente empleado por Wi-Fi u otras tecnologías ya que es luz visible. También se puede ensanchar o estrechar el haz de luz para controlar a quién están llegando los datos, fomentando así la seguridad de las comunicaciones. No obstante, el alcance es muy reducido (sobre unos 10m) y la cobertura se limita a los dispositivos que entablan visión directa con el transmisor. Incluso una mano o cualquier otro elemento podrían cortar las comunicaciones.

1.1.2 Cable KNX

Los dispositivos domóticos interconectados mediante esta tecnología son estables, seguros y muy eficientes. La implementación de una tecnología dedicada hace que la red no presente saturación ni interferencias, ya que no tiene que compartir funciones con las otras tecnologías existentes. La principal desventaja es que precisa de una importante inversión, que puede ser muy elevada si la zona a cubrir es muy extensa.

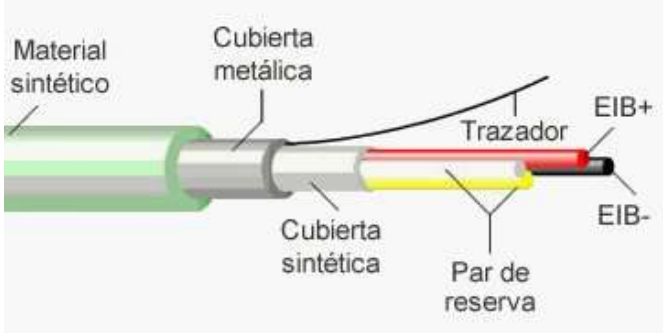

Figura 1.7 Cable KNX.

1.1.3 Cable PLC(X10)

Conocido como "powerline" utiliza el cableado de alimentación eléctrica del edificio para enviar señales. Aunque permite establecer una red domótica sin un desembolso adicional, el compartir cables con el resto de dispositivos del entorno no es óptimo, y a pesar de que se intente solventar los problemas empleando filtros, la aparición de fallos en la comunicación es muy frecuente.

1.2. Tipos de instalaciones domóticas

Aunque la domótica presenta multitud de ventajas, aun no ha conseguido dar el gran salto e introducirse en la mayoría de los hogares. Esto hace unas décadas se explicaba por la baja capacidad de procesamiento que tenían los dispositivos y la inexistencia de redes inalámbricas, sin embargo, hoy en día hemos superado ampliamente esta barrera técnica, habiendo una vasta oferta de equipos con una alta velocidad de transmisión para cualquier parte que se desee.

En la actualidad se pueden instalar multitud de elementos que podemos accionar y controlar telemáticamente mediante internet como: bombillas, interruptores, televisores, equipos de sonido, persianas, cerraduras, electrodomésticos, sistemas de climatización, riego, vigilancia remota, etc. También se pueden implementar sistemas condicionales que ejecutan una acción por si solos en base a un estímulo, señal externa u hora prefijada.

No obstante, hay un problema que todavía no se ha conseguido solventar, poner todos los elementos existentes bajo un estándar común. Observando las tecnologías más vendidas del mercado (Bluetooth, WiFi, ZigBee o Z-Wave) se comprueba que sería casi imposible interconectar todos estos equipos, y en caso de que se hiciera, el precio del hardware se dispararía.

Para obtener una interoperabilidad total, el siguiente paso en el futuro debe ser la creación de un nuevo estándar universal que pueda ser tomado por todos los fabricantes para la creación de sus dispositivos, lo que permitiría reducir enormemente la complejidad de los actuales sistemas domóticos. La complejidad actual es otra gran dificultad añadida, puesto que para crear una red se necesitan grandes conocimientos técnicos que quedan fuera del alcance de la mayoría de los usuarios, relegando así el papel de la mayoría de la población a la instalación de elementos muy sencillos como enchufes inteligentes o sistemas de iluminación.

Es previsible que si en un futuro se consigue disminuir la complejidad, la domótica empezará a ser usada de forma masiva por la población. El aumento de la demanda, traerá de la mano la reducción de costes y el aumento de la investigación y producción en este campo.

1.3. Unificación del control de dispositivos

Uno de los usos más demandados de la domótica es el control de dispositivos. Normalmente esta función se lleva a cabo mediante mandos a distancia o interfaces informáticas. No obstante, cada vez el Smartphone y tablet está tomando un papel más importante en esta función, lo cual continua con la falta de homogeneidad entre los sistemas de la red, ya que por ejemplo, los teléfonos pueden tener diferentes sistemas operativos.

En la búsqueda de nuevas formas de control de dispositivos se ha desarrollado la tecnología de control gestual, la cual mediante el movimiento de los dedos, la mano o cualquier otra parte del cuerpo permite sustituir las funciones de las viejas tecnologías de control. Pese a las bondades de esta tecnología, está teniendo poca aceptación entre el público.

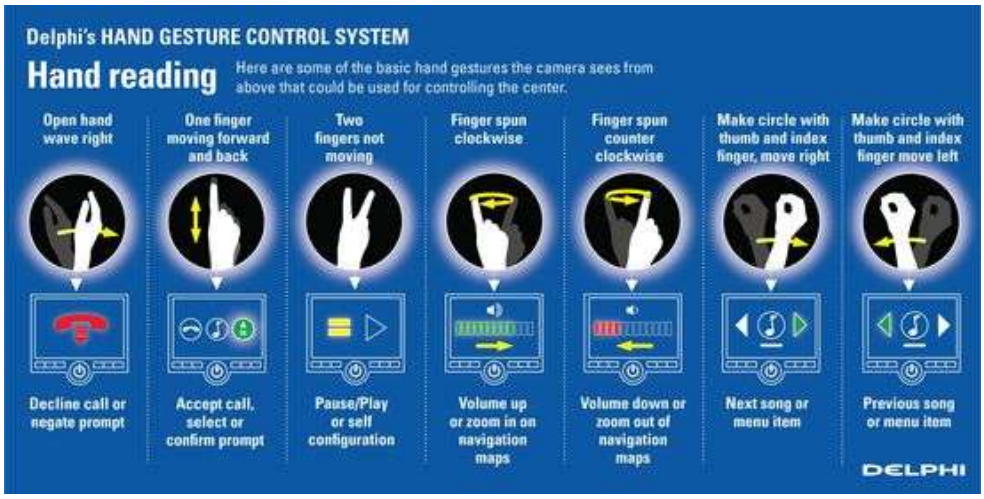

Figura 1.8 Ejemplo de diferentes señales de control gestual.

Una tecnología que está teniendo gran acogida es el reconocimiento de voz. Sistemas como Siri en iOs, Google Now en Android o Cortana en Windows ya han demostrado las ventajas de este tipo de control para la telefonía móvil, y es solo cuestión de tiempo que se implanten de forma masiva en la domótica. Amazon Echo es un buen ejemplo de este tipo de tecnología, el cual permite controlar una serie de dispositivos usando comandos de voz. Actualmente otras compañías como Apple, Samsung, Google o Microsoft están trabajando para desarrollar sus propias tecnologías.

Figura 1.9 Amazon echo.

1.4. Asistente digital centralizado

Aunque se están dando grandes pasos en el campo del reconocimiento por voz, es evidente que aún queda bastante trabajo en la tecnología encargada de reconocer el lenguaje natural. Cuando esto suceda el paso lógico será la implantación de asistentes virtuales centralizados encargados de gestionar todos los elementos conectados bajo la red. En un futuro cercano estos asistentes serán virtuales, y se comunicarán con nosotros mediante interfaces gráficas, mientras que en unos 10 o 20 años se piensa que estos asistentes podrían ser remplazados por robots.

Figura 1.10 Robot HONDA.

Cabe la duda de si estos asistentes robóticos serán capaces de adaptarse a la complejidad humana. La capacidad de desenvolverse en nuestras viviendas, comprender las distintas situaciones, y ser capaz de contextualizarlas será una ardua tarea que solo puede ir de la mano de grandes avances en inteligencia artificial.

1.5. Previsiones

Se prevé según estimaciones del estudio 'Statista Smart Home Report 2017' que el mercado de la domótica crecerá a un ritmo de 27,5% anual entre 2017 a 2022 hasta alcanzar los 112.800 millones de dólares en 2022.

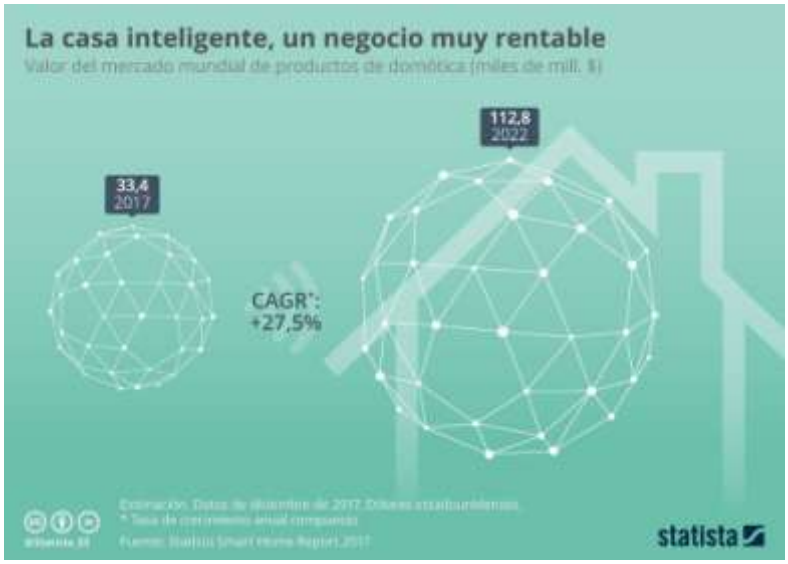

Figura 1.11 Estimación del mercado de la domótica.

1.6. Justificación del trabajo

Se parte de un montaje electrónico con el microcontrolador Arduino Uno, el cual monta un shield para internet. Este aloja en el servidor una página web accesible desde internet la cual tiene una sola función de encendido o apagado. Esta, se encargaba de controlar la activación de un relé que alimentaba un mando a

distancia de 433MHz (Figura 1.12 izquierda), emitiendo así una señal RF de 433MHz. La señal emitida es recogida por un receptor (figura 1.12 derecha) que activa un relé, accionándose así la bomba de agua.

Figura 1.12 Mando a distancia y receptor de 433MHz.

1.6.1 Limitaciones y problemas

Seguridad

La página web alojada en el Arduino protege sus funciones mediante una contraseña. Esta viaja en texto plano por la red, por lo que sería fácilmente accesible a terceros.

Comunicación simplex

Debido a que receptor y transmisor de 433MHz solo posibilitan la transmisión de información en un solo sentido, la comunicación muestra bastante inflexibilidad. Esta incapacidad, imposibilita la implantación de asentimientos con los que realizar una comunicación robusta y segura, e incluso la adquisición remota, mediante nuestro dispositivo inteligente u ordenador, de datos obtenidos en el lado receptor de la comunicación RF.

Inflexibilidad

Puesto que el lado receptor solo discrimina un mensaje, la variedad de funciones que se pueden realizar se limitan a apagar o encender.

1.6.2 Soluciones propuestas

Seguridad

Para solventar el problema de privacidad de datos se ha decidido instalar un certificado SSL en el servidor de la página web. Como no es posible albergar este certificado en un servidor con Arduino Uno, se ha tomado la decisión de sustituir el Arduino Uno por una Raspberry Pi, que es una herramienta mucho más potente la cual

puede ejercer funciones de red sin la necesidad de un shield adicional como el Arduino. Raspberry Pi presenta grandes ventajas, como una gran comunidad de usuarios y la posibilidad de trabajar sobre software libre, en concreto sobre la distribución Raspbian basada en el sistema operativo GNU/Linux.

Comunicación simplex

Para solucionar este problema se ha decidido la instalación de módulos receptor y emisor de 433MHz (Figura 1.4) en cada lado de la comunicación. Puesto que la comunicación en ambos sentidos es a la misma frecuencia, no se pueden entablar emisiones a la vez, pero si es posible la comunicación bidireccional en momentos alternos de tiempo. Esto posibilita una comunicación half-dúplex que permite la incorporación de asentimientos y la adquisición de datos en ambos lados.

Inflexibilidad

Puesto que anteriormente la comunicación era muy rígida, es decir, solo se podían hacer 2 cosas, se ha manipulado el tren de bits que manda el emisor para que una parte de este identifique al equipo que se desee manipular y otra para la información alusiva a la acción que se desea realizar con dicho equipo. Se ha logrado controlar el funcionamiento de hasta 16 equipos. Dado que ahora recibimos una información más compleja, se ha necesitado la incorporación de un Arduino Nano, el cual se encarga de controlar todos los equipos conectados a este en base a las comunicaciones recibidas por la Raspberry.

Conclusión

Con todos los cambios realizados se garantiza una comunicación segura, que salvaguarda la información de nuestro sistema. Se procurará usar en todo el proceso software libre y hardware de bajo coste a la vez que reutilizable. Con todo esto se obtendrá un sistema capaz de controlar equipos y obtener datos remotamente muy flexible, que puede ser configurado según las necesidades del usuario.

2 SOFTWARE

A lo largo de este capítulo se presentará y justificará el software que se va a emplear en el trabajo. Debido a que este trabajo pretende ser compartido con la comunidad y a que se desea desarrollar un sistema de muy bajo coste, se ha optado, dentro de lo posible, por el empleo de software libre. El código abierto permite su modificación, mejora y participación por parte de cualquier persona que quiera contribuir al perfeccionamiento del sistema.

2.1. Arduino

Arduino es una compañía que se caracteriza por la creación de productos de código y hardware abierto, conformando así una comunidad internacional que se implica en el diseño y creación de placas de desarrollo hardware para implementar dispositivos digitales que puedan controlar y detectar estímulos externos del mundo real. Así pues, desde su nacimiento, Arduino se enfoca tanto en acercar la electrónica y facilitar el uso de los microcontroladores, como en el desarrollo de sistemas embebidos en proyectos de diversa índole.

Los productos que lanza al mercado la compañía se distribuyen como Software y Hardware libre, bajo la Licencia Pública General Reducida de GNU (LGPL) o la Licencia Pública General de GNU (GPL), siendo posible la fabricación de placas y la distribución de su software por parte de cualquier individuo.

Figura 2.1 Logo de Arduino.

El software de Arduino está formado por dos elementos: el entorno de desarrollo (IDE), y en el cargador de arranque (bootloader) que se ejecuta automáticamente dentro del propio microcontrolador desde el momento en el que se alimenta.

Las placas de Arduino se programan mediante un ordenador, usando comunicación serie.

2.1.1 Arduino IDE

Es un entorno de desarrollo integrado disponible para Windows, macOS y Linux escrito en el lenguaje de programación Java. Se usa para subir código a las diferentes placas de Arduino, aunque también se puede subir código a placas proporcionadas por otros vendedores.

El código fuente del IDE está disponible bajo GNU (General Public License), versión 2. El entorno soporta los lenguajes C y C++ con sus respectivas reglas de estructurado de código, los cuales componen las partes del programa o sketch.

El IDE de Arduino emplea el programa avrdude para convertir el código ejecutable a un archivo de texto codificado en hexadecimal que se ha subido a la placa mediante el loader que se encuentra en el firmware.

Figura 2.2 IDE de Arduino.

Sketch

Para la programación del sketch, que es como se denomina a un programa de Arduino, se emplea una adaptación de C++ proveniente de avr-libc que no es más que una librería de alta calidad que se puede usar con GCC (compilador de C y C++) en los microcontroladores de Atmel.

No obstante, aunque se hable de un lenguaje de programación propio, esto no es del todo cierto, ya que realmente parte de la base de C++ pero añade una serie de librerías denominadas core que facilitan la manipulación y la ejecución de tareas específicas sobre los pines de entrada/salida. El IDE incluye todas estas librerías de forma automática y no es necesario declararlas.

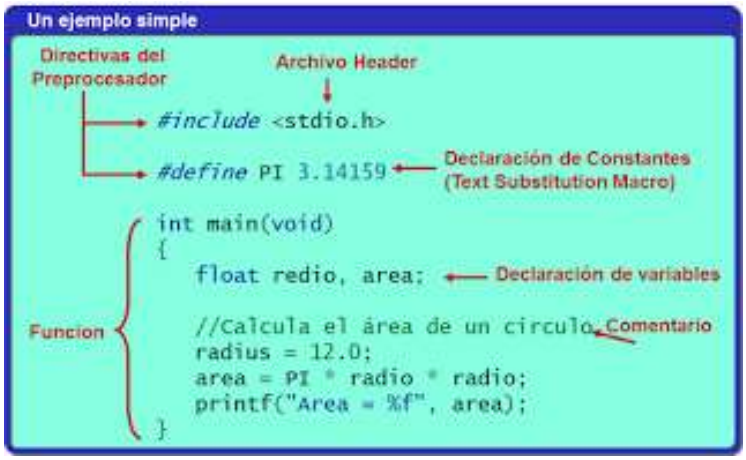

Figura 2.3 Estructura básica de un sketch de Arduino.

2.1.2 Bootloader

Cuando cargamos mediante el IDE un programa en Arduino, realmente estamos empleando el bootloader, que no es más que un pequeño programa que ha sido cargado con antelación en el microcontrolador de la placa, el cual es capaz de cargar el código que deseemos sin necesidad de adquirir hardware adicional. El bootloader solo está activo unos segundos cuando se resetea el Arduino, continuando luego con la ejecución del sketch que está en la memoria flash.

El bootloader siempre se ejecuta cuando el microcontrolador se alimenta o se pulsa reset. Durante un pequeño tiempo definido espera para comprobar que no le llegue un nuevo sketch por el puerto serie. Este evento puede ser distinguido sin problema, ya que el sketch tiene un tipo definido. En el caso de que recibiese un nuevo sketch, lo ejecutaría tras almacenarlo en la memoria flash, pero si no se recibiese nada, se ejecutaría el que anteriormente se había cargado en la placa.

La memoria flash del ATMega328p alberga el bootloader, el cual ocupa 0,5KB de los 32 que hay disponibles. Este bootloader suele venir precargado de fábrica.

Para la comunicación con el bootloader se utiliza el protoclo STK500 propio de Atmel.

Figura 2.4 Microcontrolador ATMega328p SMD de la marca Atmel.

2.2. KiCad

Figura 2.5 Logo de KiCad.

KiCad es un paquete de software libre creado para facilitar el diseño electrónico automáticamente. Básicamente es una herramienta que facilita el desarrollo de circuitos esquemáticos y los convierte a placas de circuito impreso (PCB).

KiCad está organizado en tres partes:

- kiCad- Administra el proyecto.
- eeschema- Edita los componentes y conexiones del esquemático.

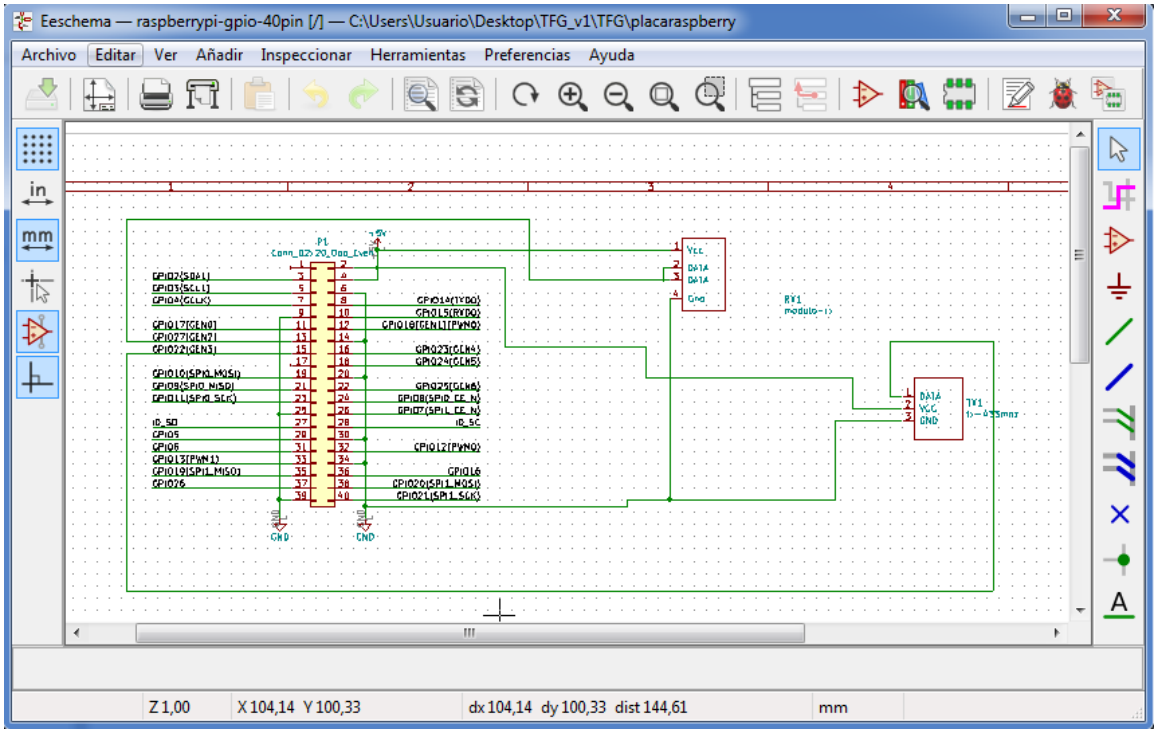

Figura 2.6 eeschema de la PCB para la Raspberry Pi.

- gerview-Visualiza los archivos Gerber. Este no es más que un tipo de formato de archivo que contiene toda la información necesaria para la fabricación de la placa. En resumen, es el archivo que se debe proporcionar al fabricante para que fabrique la placa que se desea.

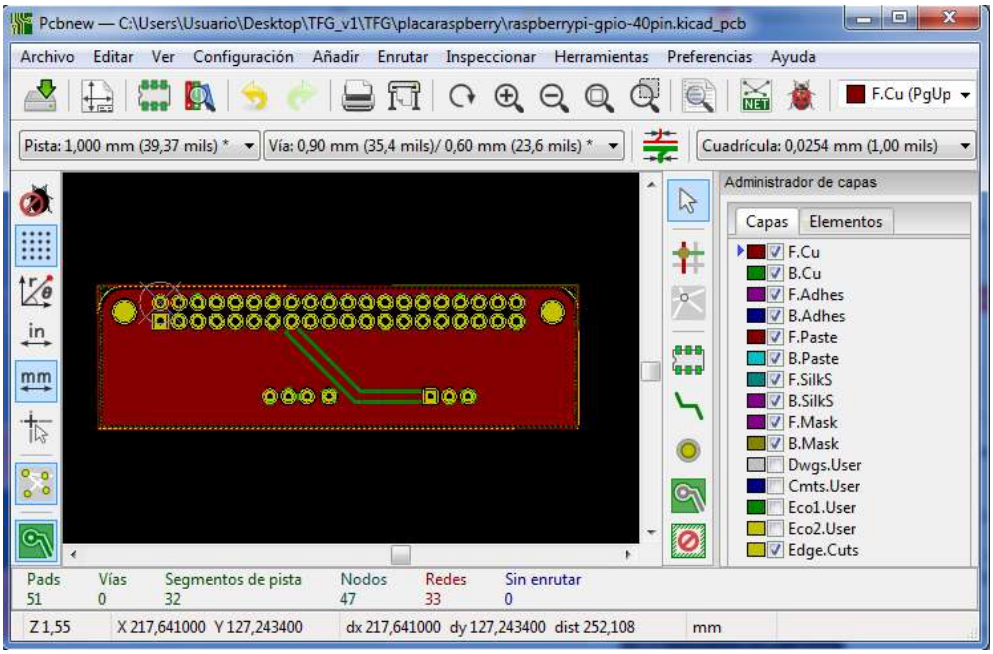

Figura 2.7 gerview de la PCB para la Raspberry Pi.

2.3. VNC

VNC (Virtual Network Computing) es un programa de software libre con una estructura cliente/servidor que permite visualizar remotamente las acciones de un equipo servidor en la pantalla de otro equipo cliente. Además de la visualización, VNC permite operar sin ningún tipo de restricción sobre el equipo que se desea controlar. Este programa será de gran utilidad, ya que como se verá a continuación en más detalle en la sección hardware, la Raspberry Pi no dispone de una pantalla sobre la que visualizar su contenido o las acciones que se ejecutan sobre ella.

Un sistema de VNC está compuesto por tres elementos:

- El servidor de VNC es el programa que está instalado en el equipo que está compartiendo su pantalla.
- El cliente VNC es el programa que está instalado en el equipo que se usa para visualizar la pantalla del equipo servidor.
- El protocolo VNC (RFB) se basa en una primitiva gráfica del servidor hacia el cliente y mensajes de eventos en el sentido opuesto.

2.4. SD Card Formatter

Este software se encarga de proporcionar una herramienta muy sencilla y rápida para borrar todo el contenido almacenado en una memoria SD. Este procedimiento es necesario antes de instalar Raspbian en la tarjeta microSD que vaya a usar la Raspberry Pi.

2.5. Win32DiskImager

Es una aplicación de código abierto que graba imágenes de CD o DVD en memorias USB o tarjetas SD,

siendo así un lector de discos virtual. Por este motivo, el programa se usará para grabar la imagen de Raspbian en la tarjeta microSD. También es posible generar imágenes leyendo los datos almacenados en la memoria, lo cual resulta de gran utilidad para la creación de copias de seguridad para Raspbian.

2.6. PuTTy

PuTTy es un cliente SSH, Telnet, rlogin y TCP raw con licencia libre. En este proyecto será usado para inicializar las funciones necesarias en Raspbian para poder acceder a esta mediante VNC, y por tanto a la interface gráfica que se ha instalado.

2.7. Fritzing

Es un software libre que permite crear una representación visual de circuitos electrónicos, facilitando las labores de documentación y distribución de montajes electrónicos de forma clara y concisa entre la comunidad de usuarios.

2.8. GNU/LINUX

GNU/Linux, conocido popularmente como Linux es un sistema operativo libre de tipo Unix. Se caracteriza por ser multiplataforma, multiusuario y multitarea. El sistema nace de la integración de distintos trabajos, entre los cuales cabe destacar el encabezado por Richard Stallman y la Free Software Foundation, y Linus Torvalds al frente del núcleo Linux. El gran atractivo de este SO, y lo que lo hace uno de los más extendidos en el mundo de software libre, es que su código fuente puede ser utilizado, modificado e incluso redistribuido libremente por cualquier persona o entidad bajo la Licencia Pública General (GNU) y otras licencias libres.

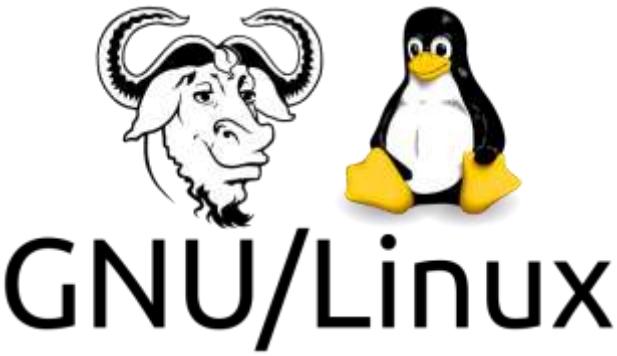

Figura 2.8 Logo GNU/Linux.

Normalmente GNU/Linux se encuentra en forma de agrupaciones que se conocen como distros o simplemente distribuciones, a las que simplemente se les ha añadido una serie de aplicaciones y programas para descargar e instalar. El objetivo es presentar GNU/Linux como un producto final que el usuario tiene disponible para su utilización inmediata.

Pese a la distribución libre de GNU/Linux, este posee la cuota más importante del mercado en aplicaciones concretas como servidores de Internet, supercomputadoras y sistemas embebidos. De acuerdo informes de IDC y Top500.org, GNU/Linux es empleado por el 78% de los principales 500 servidores del mundo. Aunque el sistema está disponible para otros dispositivos como ordenadores personales, teléfonos inteligentes, video consolas y otros dispositivos, no goza de tanta popularidad.

2.8.1 Raspbian

Raspbian es una distribución del sistema operativo GNU/Linux y por lo tanto libre, que se basa en Debian Stretch para Raspberry Pi.

Al ser una distribución libre, la versatilidad y opciones que brinda Raspbian son muy variadas. Esta distribución como la gran parte de las pertenecientes a GNU/Linux contiene repositorios que pueden ser descargados e instalados libremente. Por este motivo, Raspberry Pi a parte de funcionar como un microcontrolador, puede hacer las funciones de ordenador personal, todo ello a un coste muy reducido.

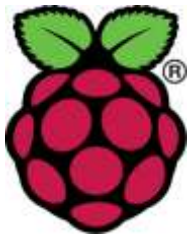

Figura 2.9 Logo de Raspbian.

3 HARDWARE

En el desarrollo de este trabajo se ha empleado una serie de elementos físicos los cuales van a ser descritos en este capítulo. También se pretende justificar el uso de cada componente, siendo especialmente cuidadosos en no disparar el coste. Como todo el software usado es gratuito, el precio total de todo este proyecto recaerá sobre el Hardware. A modo orientativo, para que el lector se pueda hacer una idea anticipada, se puede observar la figura 3.1 para tener una idea global del proyecto.

Figura 3.1 Esquema global del proyecto.

3.1. Raspberry Pi

Raspberry Pi es esencialmente un ordenador de placa reducida o placa simple de bajo coste desarrollado en el Reino Unido con el objetivo de estimular la enseñanza informática en las aulas de una forma asequible.

Aunque es una propiedad registrada, Raspberry permite su uso libre tanto a nivel educativo como personal, pudiendo cualquier persona convertirse en revendedor o redistribuidor de sus tarjetas. Es decir, a efectos prácticos, Raspberry es un hardware libre, aunque la empresa reserve su derecho.

El modelo empleado para el proyecto es la Raspberry Pi 3B+ tiene las siguientes especificaciones:

- Procesador Broadcom 1.4GHz 64-bit quad-core ARMv8.
- Memoria RAM.
- GPU Broadcom VideoCore IV 400 MHz.
- 4 puertos USB.
- Puerto HDMI y Ethernet.
- 40 pines GPIO.
- Conector para cámara.
- No incluye memoria, siendo necesaria la introducción de una tarjeta microSD en las placas más modernas.
- Consumo energético de 800mA, 4W.

- Conectividad Wifi 802.11n/ac, Bluetooth 4.2 BLE.
- Dimensiones de 85,6mm x 53,98mm.
- Fuente de alimentación de 5V vía Micro USB o GPIO.

Figura 3.2 Partes de la placa Raspberry Pi modelo 3 B+.

3.1.1 Mejoras de Raspberry Pi 3 Modelo B+

Salió a la luz en marzo de 2018 para actualizar el anterior modelo B y entre sus mejores se incluye un nuevo procesador (el modelo B pasa de tener 1,2GHz a tener 1,4GHz) y una mayor conectividad, que se caracteriza por la posibilidad de usar las bandas de 2,4GHz y 5GHz, las cuales son muy habituales en Wi-Fi. El puerto Ethernet triplica su tasa binaria, pasando de los 100 a los 300Mbit/s.

3.1.2 Comunidad

Una de las razones para la elección de esta placa es la extensa comunidad que hay tras de ella, facilitando las labores de programación y depurado de errores, así como la instalación de librerías y programas.

3.1.3 Conclusiones

Se ha elegido este sistema por las siguientes razones principales:

- Software libre.
- Bajo coste, aproximadamente 37€.
- Integración de Wi-Fi y puerto Ethernet, lo cual es ideal para su uso como servidor.
- Posee pines GPIO que permiten usarlo como un microcontrolador clásico.
- Posibilidad de albergar servidores gratuitos como Apache.
- Interface que hace más intuitiva la programación.
- Posibilidad de instalar certificados de cifrado punto a punto como SSL.
- Gran capacidad de computación.
- Tamaño reducido.

- Gran número de usuarios que componen la comunidad Raspberry.

3.2. Arduino nano

Es un microcontrolador desarrollado por Arduino y basado en el chip Atmega328p/Atmega168. Las placas de Arduino están muy extendidas en robótica, sistemas embebidos y proyectos de electrónica donde la automatización de procesos es esencial.

A diferencia de en otros microcontroladores, cualquier tipo de soporte técnico está disponible en la comunidad de Arduino, en la cual se comparte contenido de forma gratuita y desinteresada.

Tabla 3.1 Pines del Arduino nano.

Pin	Uso
D0-D13	Entrada Digital / Pines de Salida
A0-A7	Entrada Analógica / Pines de Salida
0 (RX), 1 (TX)	Pines de comunicación serie
D10,D11,D12,D13	Pines de comunicación SPI
A4,A5	Pines de comunicación I2C
D2,D3	Interrupciones externas

3.2.1 Características

El Arduino nano es un microcontrolador desarrollado por Arduino.cc en Italia basándose en el ATmega328p (el que se usará en el proyecto) o ATmega168.

Tiene exactamente las mismas funciones que el Arduino Uno, siendo totalmente operable el mismo código en ambas placas, pero con un tamaño más reducido.

Entre sus características principales destaca:

- El voltaje de operación es de 5V, aunque se le puede entregar un voltaje de 7 a 12V.
- La placa contiene 14 pines digitales, 8 analógicos, 2 resets y 6 pines de alimentación.
- Los pines analógicos tienen una resolución total de 10 bits los cuales sirven para medir un valor de 0 a 5V.
- Viene con un oscilador interno de 16MHz.
- El conector usado para programación y/o alimentación es el Mini USB.
- Memoria Flash de 16KB(Atemag168) o 32KB(ATmega328p). Esta memoria es usada para almacenar el código, estando 2KB reservados para el almacenamiento del bootloader.

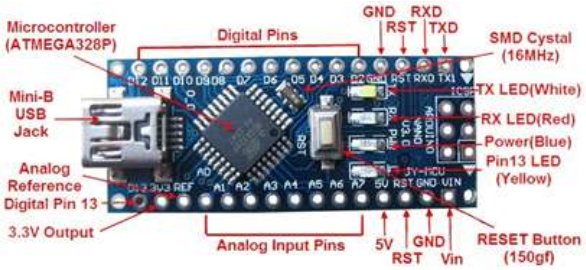

Figura 3.3 Partes del Arduino Nano.

3.2.2 ATmega328p

Es un circuito integrado de alto rendimiento que está basado en el microcontrolador RISC, combinando la antes citada memoria Flash de 32KB, la cual permite la lectura y escritura al mismo tiempo, con:

- Memoria EEPROM de 1KB.
- SRAM de 2KB.
- 23 Líneas de entrada/salida de propósito general.
- 32 registros de propósito general.
- 3 temporizadores/contadores con modo de comparación.
- Interrupciones externas e internas.
- Conversor A/D de 10 bits.
- Puerto serie SPI.

Figura 3.4 ATmega 328P SMD.

3.2.3 Comunidad

El factor que más ha contribuido al éxito de Arduino es la gran comunidad que respalda al proyecto, publicando día a día nuevo contenido, divulgando y respondiendo a dudas de una forma totalmente altruista. Toda la información necesaria está disponible en Internet, donde hay todo tipo de cursos, tutoriales, herramientas, foros, proyectos compartidos por otro usuario…

3.2.4 Conclusiones

Debido a la gran comunidad que respalda Arduino, las dimensiones reducidas, la potencia de computo, y el bajo precio (1,3€), gracias a la posibilidades que ofrece el hardware libre, Arduino nano es el microcontrolador

adecuado para el proyecto.

3.3. Tarjeta microSD

Constituye la memoria para el modelo de Raspberry Pi 3B+, entre otros.

Tiene un tamaño de 15x11x1mm, con un área de 165 mm^2. El régimen binario no es muy alto, sin embargo empresas como SanDisk han alcanzado hasta los 98MB/s.

Para este proyecto se aconseja adquirir una tarjeta microSD de más de 8GB y con una tasa binaria de más de 10MB/s que no lastre el sistema. Para este proyecto se ha usado la tarjeta que se puede visualizar en la figura x, la cual tiene 32GB y una tasa de 80MB/s.

Figura 3.5 Tarjeta microSD.

3.4. Punto de acceso

Un punto de acceso inalámbrico, también conocido popularmente como router, es un dispositivo de red usado para interconectar equipos como ordenadores, smartphones o tablets a la red de forma inalámbrica.

Su función no es solo interconectar equipos en una red personal, sino que también hace las veces de intermediario entre los equipos de la red e Internet u otra red.

El gran atractivo de los puntos de acceso es que facilita la interconexión de nuestros equipos de la red privada sin usar cables, ofreciéndose así un sistema menos aparatoso y sencillo de mantener. Al ser un dispositivo ya existente en la mayoría de los hogares, permite aprovechar los recursos existentes y rebajar el coste del proyecto.

3.5. Módulos transmisor y emisor de 433MHz

Descritos en la sección 1.1.1, se han elegido para la elaboración de este proyecto debido los siguientes motivos:

- Coste muy reducido (1,2 € el par receptor/emisor).
- Posibilidad de modular las emisiones, lo que permite codificar el mensaje según se desee.
- Menor presencia de interferencias frente a otros sistemas como Bluetooth que operan en bandas ya muy saturadas.
- Mayor alcance debido al mejor comportamiento frente a obstáculos que ofrece una longitud de onda mayor.
- Posibilidad de que los módulos sean interoperables con el sistema del que se parte.

3.6. Antena monopolo para el módulo de 433MHz

Para el proyecto se va a introducir un antena monopolo vertical por cada módulo transmisor/receptor ya que aumentan considerablemente el alcance de transmisión como se muestra en la tabla 3.2, disminuyen la necesidad energética en los módulos y son muy baratas y fáciles de construir, ya que son esencialmente un cable de cable de cobre cortado con la dimensión correcta.

Tabla 3.2 Tabla de alcances.

Emisor	Receptor	Alcance
Sin antena	Sin antena	3m
Sin antena	Con antena	10m
Con antena	Sin antena	10m
Con antena	Con antena	100m

Para el cálculo de la longitud se ha de tener en cuenta que esta ha de ser igual a $\frac{\lambda}{4}$, por lo tanto la longitud quedará de la siguiente forma:

$$L = \frac{\lambda}{4} = \frac{\frac{c}{f}}{4} = \frac{\frac{3\times10^8}{443\times10^6}}{4} = 0.169m = 16.9cm \quad (3.1)$$

3.7. Cable RJ45

El cable RJ45 es usado en este proyecto para establecer una conexión física entre el ordenador y la Raspberry Pi, ya que esta, como se ha resaltado ya varias veces, no dispone de pantalla ni teclado con la que poder configurar todos sus parámetros para hacerla operativa.

Posee ocho pines con cables de par trenzado en sus extremos. El uso más común, y para el que se requiere en este proyecto, es el de cable de red Ethernet.

Figura 3.6 Cable de red RJ45.

3.8. Fuente de alimentación

Es un dispositivo encargado de convertir la corriente alterna en una fuente de corriente continua constante para alimentar equipos electrónicos. En este proyecto se empleará para alimentar los equipos a controlar cuando la alimentación aportada por el Arduino Nano sea insuficiente o bien como fuente única de alimentación, en caso de que no se esté alimentando el Arduino mediante el cable mini USB. Estas dos funciones se podrán llevar a cabo mediante el control de un jumper existente en la placa.

3.9. Sensores y equipos a controlar

En esta sección se detallarán todos los equipos que se serán controlados mediante el sistema de este proyecto. Es decir, como objetivo último, estos son los dispositivos que tienen la funcionalidad real del trabajo.

3.9.1 Relé

El relé es un dispositivo electromagnético el cual hace las veces de interruptor. Es controlado mediante un circuito eléctrico, un Arduino nano en este proyecto, el cual mediante una bobina y un electroimán acciona uno o varios contactos que permiten alimentar un dispositivo eléctrico independiente.

En general, el relé es usado para controlar la alimentación de dispositivos de mucha mayor potencia como bombas de agua o motores mediante dispositivos electrónicos de baja potencia, como es el caso de Arduino.

Figura 3.7 Relé.

3.9.2 Servomotor

Un servomotor o servo es un dispositivo actuador que se caracteriza por posicionar su eje en un ángulo determinado dentro de su rango de operación, y de asegurar la estabilidad de la posición. Está compuesto por un motor de corriente continua, una caja reductora y un circuito encargado de controlar el movimiento. Normalmente los servos se limiten a giros de 0 a 180º, aunque existen servos en el mercado que ejecutan una vuelta completa.

Los usos más frecuentes de los servos se dan en los campos de radiocontrol y en robótica, siendo su principal tarea posicionar un elemento físico en una posición determinada, garantizando la estabilidad y la exactitud de la posición.

El circuito de control de ubicación consiste en un controlador proporcional, el cual establece un punto de referencia, que es el deseado, mediante una señal de control cuadrada. La posición del servo se indica por codificación de ancho de pulso, es decir, el servo estima el ángulo en base al tiempo que está la señal de control a nivel alto o bajo durante un ciclo de trabajo.

En primer lugar, un amplificador de error calcula el error de posición actual, el cual es igual a la diferencia entre la referencia o posición deseada y la posición actual. Conforme a mayor sea el error, se moverá más rápidamente el motor hasta obtener un error de posición igual a 0.

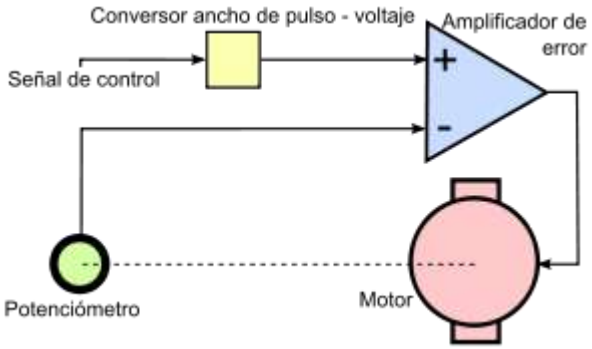

Figura 3.8 Diagrama de bloques del control proporcional de un servomotor.

Puesto que el amplificador de error debe restar dos valores analógicos, la señal de control modulada por ancho de pulso se convierte a un valor analógico de voltaje mediante un conversor. El valor actual de la posición del motor es calculado usando un potenciómetro de realimentación acoplado mecánicamente a la caja reductora del eje motor, es decir, cuando el servo rote, el potenciómetro también lo hará, variando de esta forma el voltaje introducido en la comparación.

Figura 3.9 Microservo.

3.9.3 DHT11

Es un sensor digital de temperatura y humedad del aire, el cual posee las características que se describen en la tabla.

Tabla 3.3 Características del sensor DHT11.

Medida	Valor
Temperatura(ºC)	0 a 50, con una precisión de 2ºC
Humedad del aire (%)	20-80%
Frecuencia de muestreo	1Hz

Figura 3.10 Sensor DHT11.

3.9.4 Sensor de humedad del suelo

El sensor de humedad del suelo empleado en este proyecto funciona midiendo la constante dieléctrica o permitividad del suelo existente entre las dos puntas de la sonda. Conforme a esta medición el sensor devuelve al microcontrolador una medida analógica proporcional al nivel de humedad.

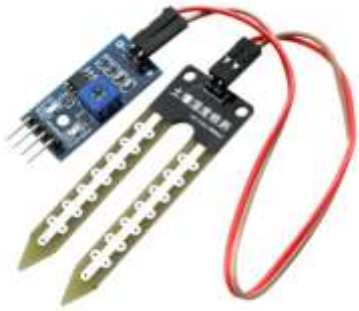

Figura 3.11 Sensor de humedad del suelo.

3.10. Elementos para el montaje electrónico de prototipado

El fin de este proyecto es dejar un sistema funcional lo más compacto posible. Para ello se realizará un sistema embebido en el que se integrarán todos los componentes electrónicos necesarios. No obstante, antes de comenzar a desarrollar el sistema embebido, se ha de probar todo el montaje electrónico en una placa experimental o protoboard, la cual permite cambiar fácilmente las conexiones entre los componentes hasta que se llegue a un sistema final totalmente comprobado y testado, cuyo diseño pueda ser fabricado en placa PCB.

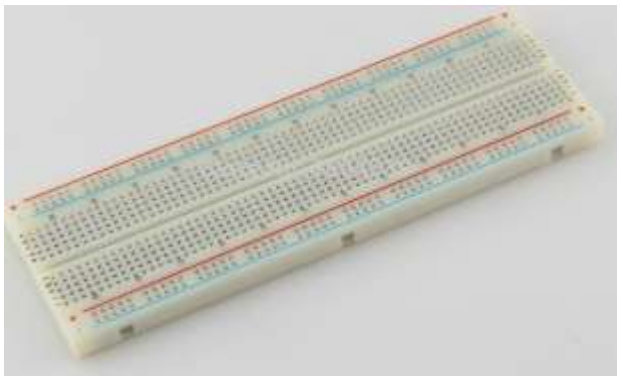

Figura 3.12 Protoboard.

3.10.1 Protoboard

Una placa de pruebas o protoboard es una placa con orificios en la que se pueden insertar cables para el conexionado o componentes electrónicos sin la necesidad de realizar soldaduras u otro tipo de conexiones más complejas. Los orificios de la placa están distribuidos por grupos los cuales se encuentran conectados independientemente del resto siguiendo un patrón determinado como por el ejemplo el seguido en la figura. Los orificios de un mismo grupo, o nodo como se le conoce en el campo de la electrónica, están interconectados por material conductor como por ejemplo el cobre.

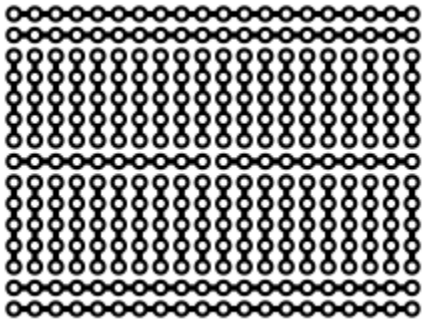

Figura 3.13 Interconexión de los orificios de la placa.

3.10.2 Cables Puente

Es un cable con un conector macho o hembra en cada punta que se usa para transferir señales eléctricas de un punto a otro de la placa de pruebas. La conexión es muy sencilla, basta con introducir una conector macho en un orificio de la placa.

Figura 3.14 Cable puente hembra-macho.

3.11. Implementación en protoboard

Dado que mediante una foto no se podría mostrar el conexionado con claridad, se ha recurrido al programa fritzing.

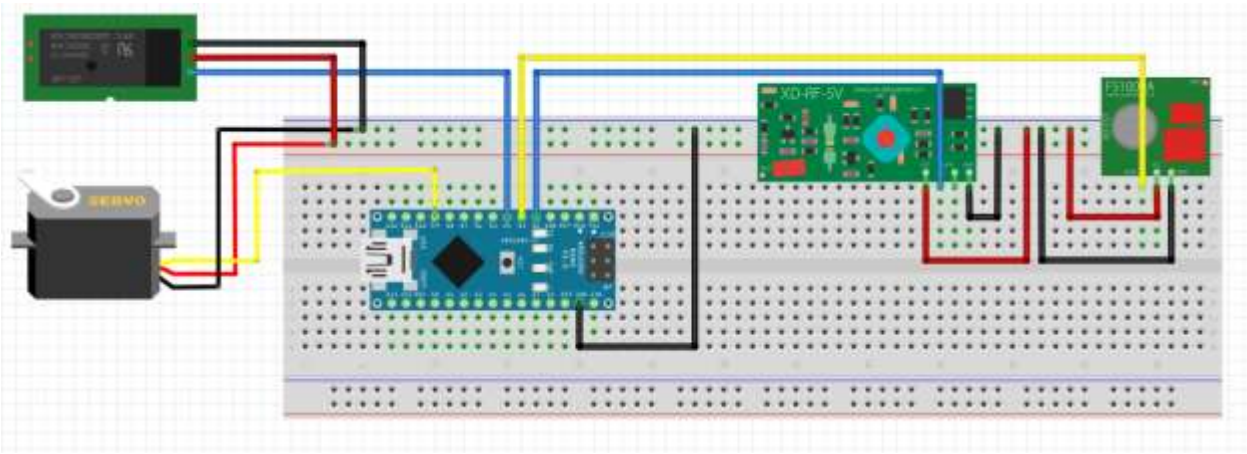

Figura 3.15 Montaje experimental de Arduino

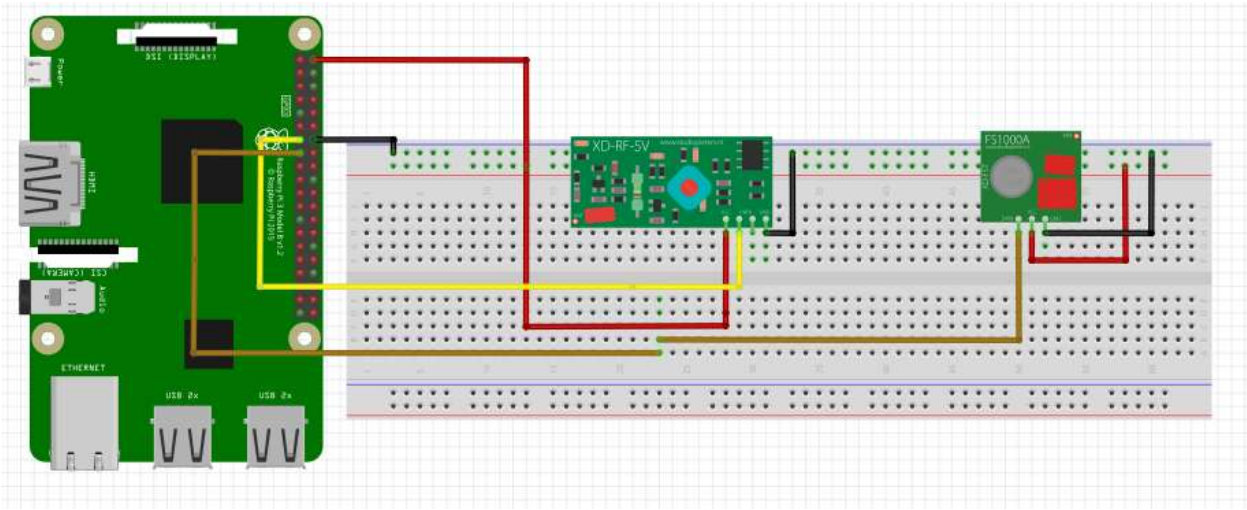

Figura 3.16 Montaje experimental de Raspberry Pi.

3.12. Diseño de las placas PCB

Una vez que todo el montaje experimental funciona de acuerdo a lo esperado, se puede proceder a la creación de unas placas PCB que hacen que el sistema quede más compacto y robusto frente a malas conexiones de los cables puente.

Como se ha comentado en secciones anteriores, el programa elegido para esta tarea es KiCad, el cual permite crear en primer lugar un esquemático o eeschema que contiene todos los componentes y conexiones existentes en el sistema como se puede observar en las figuras X y Y.

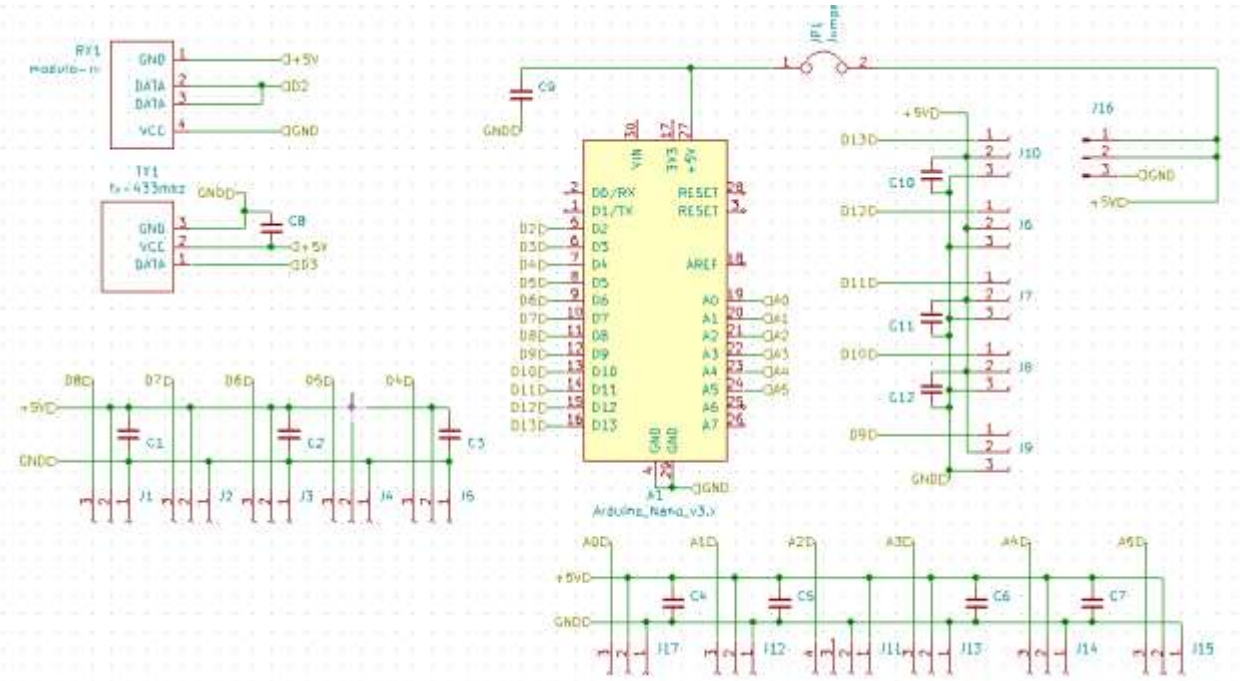

Figura 3.17 Eeschema de Arduino.

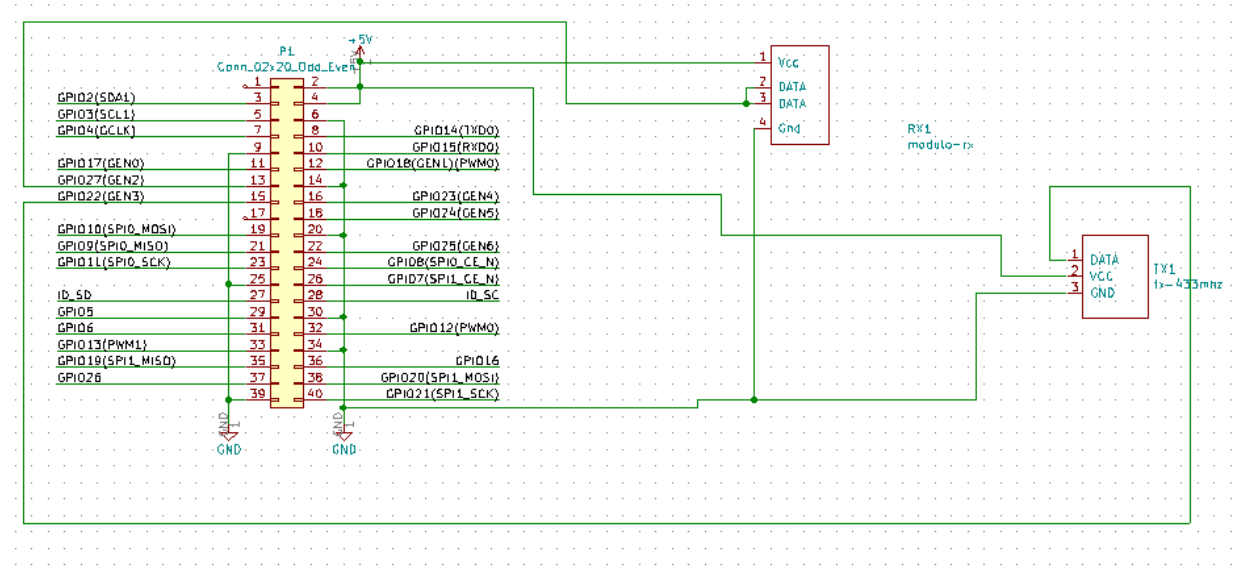

Figura 3.18 Eeschema de Raspbery Pi.

Partiendo de estos esquemáticos, a los cuales hay que asociarles una huella para la futura PCB, se construyen las placas enrutando las diferentes pistas conductoras que conexionan los componentes. Puesto que la placa va a ser manufacturada por una empresa especializada se puede añadir una serigrafía, que sirve para esclarecer las diferentes partes de las placas.

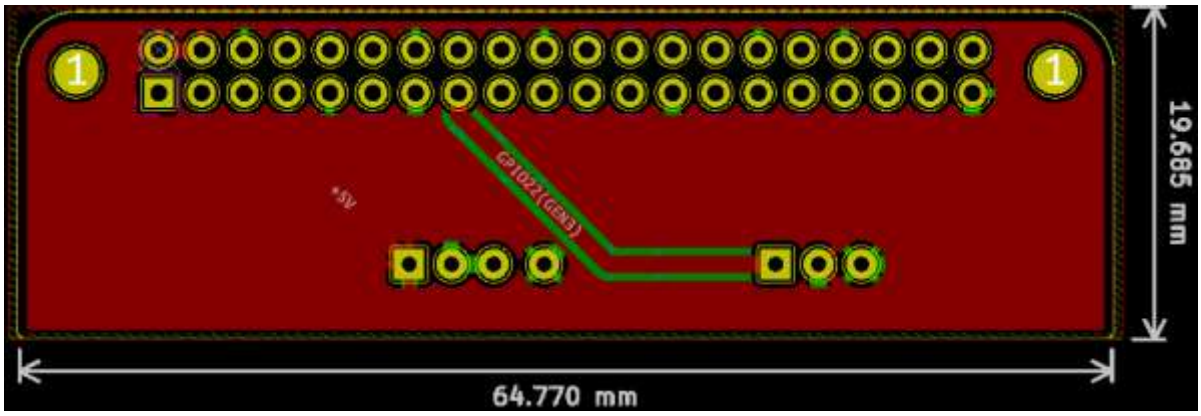

Figura 3.19 Gerview de Raspberry Pi.

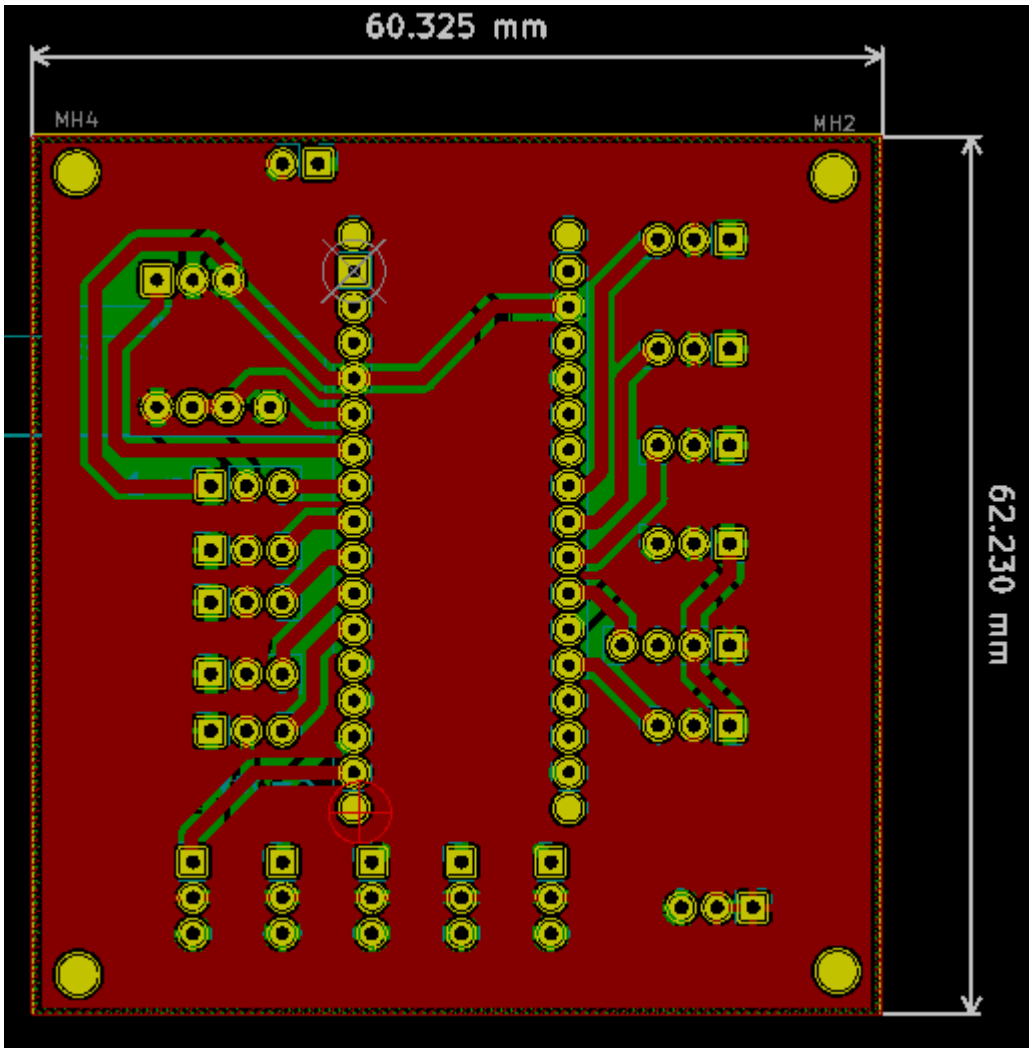

Figura 3.20 Gerview de Arduino.

Este diseño sería enviado al fabricante obteniendo el resultado que se puede ver en la figura X.

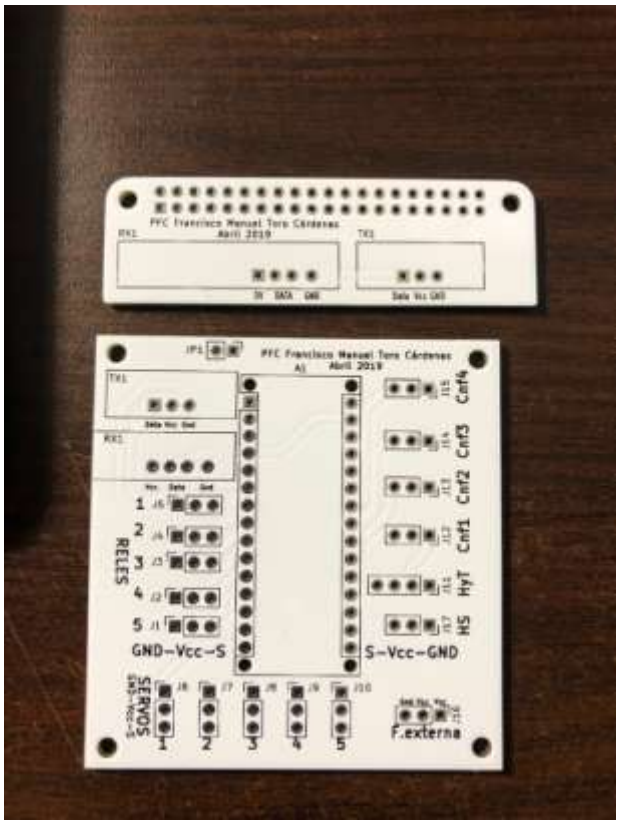

Figura 3.21 Placas PCB del sistema.

Ya se pueden soldar los componentes y encajar todos los elementos, obteniéndose el producto final.

Figura 3.22 Raspberry Pi con el shield diseñado.

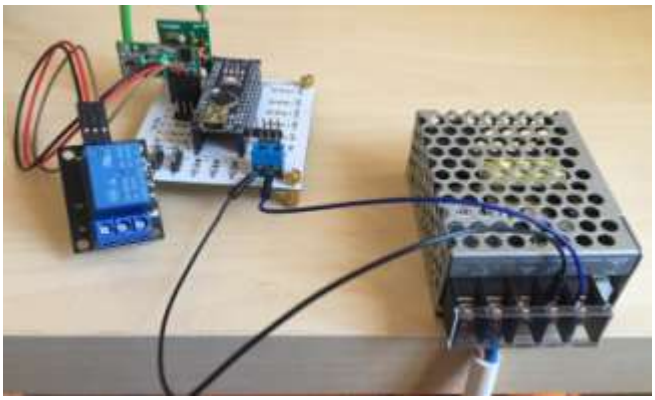

Figura 3.23 Fuente de alimentación y PCB con todos los componentes.

4 CONFIGURACIÓN DE LA RED

En la siguiente página web se puede descargar la imagen que contiene Raspbian Stretch con escritorio y software adicional recomendado. Uno de los programas adicionales de esta opción es el VNC, que como ya se ha explicado en la sección 2.3 será de gran utilidad para la programación de la Raspberry Pi. Aunque existen otras dos versiones de Raspbian mucho más ligeras y sencillas, se ha elegido esta al ser la que programa un entorno de programación más sencillo e intuitivo para el usuario medio, es decir, un escritorio con una interface que te permite ejecutar tareas sin la necesidad de usar la consola de comandos.

```
https://www.raspberrypi.org/downloads/raspbian/
```

Para la instalación de Raspbian y preparación del sistema para el uso de VNC será necesario realizar los siguientes pasos ordenadamente:

- Una vez formateada la tarjeta SD con el programa SDCardFormatter, se deberá proceder a grabar la imagen de Raspbian Stretch con el programa Win32DiskImager en la tarjeta.
- Para facilitar la tarea en los primeros pasos de configuración de la Raspberry, se procederá a asignar una IP estática al adaptador Ethernet de la Raspberry, la cual no variará en la multitud de reinicios que necesitará el sistema. Se ha de recalcar que esta IP asignada será para una comunicación Ethernet porque las primeras conexiones que se realizarán con la Raspberry serán a través de una conexión física (cable RJ-45) en la red local. Para ello se abrirá con el editor de textos el archivo almacenado en la tarjeta SD llamado cmdline y se añadirá justo al final una línea siguiendo el formato descrito a continuación : ip=IP_rpi::IP_gatewey:Máscara_sub_red:dispositivo:eth0:off. En el sistema de este proyecto dicho línea tendrá la siguiente apariencia.

  ```
  ip=192.168.137.2:::255.255.255.0:rpi:eth0:off
  ```

- Paralelamente se debe modificar el adaptador de red Ethernet en el ordenador. En resumen, se debe acabar con el esquema de red descrito en la figura siguiente.

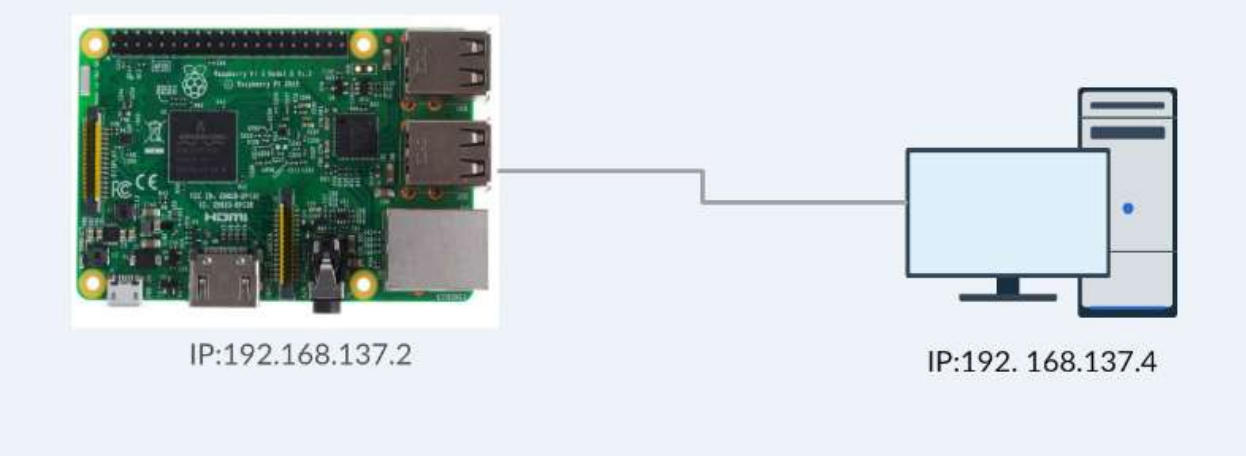

Figura 4.1 Esquema de conexión de red.

- A continuación se debe inicializar el servicio SSH de la Raspberry Pi creando un documento de texto sin extensión y vacío llamado ssh en la tarjeta SD. Esto permitirá entablar la comunicación cliente/servidor mediante el programa PuTTy. Para realizar este paso se deberá introducir la tarjeta SD en la Raspberry y conectar el cable de red Ethernet. Una vez comprobada la conectividad de red entre el ordenador y Raspberry, procedemos a abrir el programa PuTTy, ajustando los parámetros como se refleja en la figura 4.2. Tras este último paso solo quedaría introducir usuario y contraseña, que por defecto son pi y raspberry respectivamente.

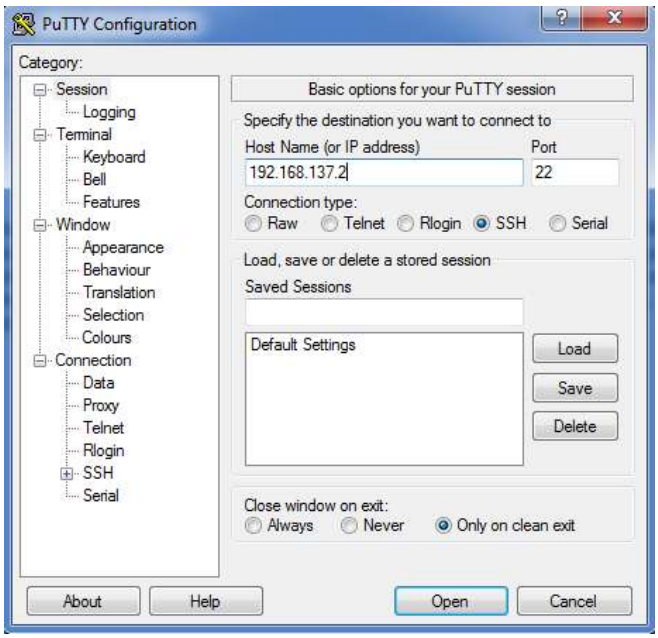

Figura 4.2 Configuración de PuTTy.

- Introducción del comando siguiente en la consola de PuTTy.

```
$ sudo raspi-config
```

- En el menú desplegable que aparece deberán realizarse 2 acciones: primero deberemos cambiar la contraseña por motivos de seguridad, y segundo se procederá a habilitar el servidor VNC. Esta última acción se encuentra en la opción Interfacing Options. Una vez dentro de esta opción se puede aprovechar para activar la manipulación de los pines de la Raspberry (GPIO) remotamente.
- Para concluir, se abre el cliente VNC y se entabla la conexión con la Raspberry indicando la IP que se está usando para la conexión local Ethernet. Posteriormente se podría configurar opciones como: lenguaje, zona horaria, conexión de red inalámbrica…

Figura 4.3 Visualización del escritorio de Raspberry Pi mediante VNC.

Instalación y configuración de Apache Server

El servidor HTTP Apache es un servidor de código abierto diseñado para las plataformas Unix, Microsoft Windosw, Macintosh y otras, el cual usa el protocolo HTTP/1.1 y la noción de sitio virtual según la normativa RFC 2616.

Apache al ser un código abierto ofrece muchas opciones de configuración, a la vez que proporciona bases de datos de autentificación y negocio.

Para instalar el servidor Apache en la Raspberry Pi debemos seguir los siguientes pasos:

- Primero se ha de crear el grupo de Internet www-data mediante los siguientes comandos:

```
$ sudo groupadd www-data
$ sudo usermod -a -G www-data www-data
```

- Seguidamente se deben actualizar los repositorios y los programas de la Raspberry.

```
$ sudo apt-get update
$ sudo apt-get upgrade
```

- Ahora se procede a la instalación de Apache.

```
$ sudo apt-get install apache2
```

Una vez instalado, se puede comprobar que todo ha ido correctamente introduciendo en el navegador de Raspberry su propia ip y visualizando la página que tiene el servidor Apache por defecto.

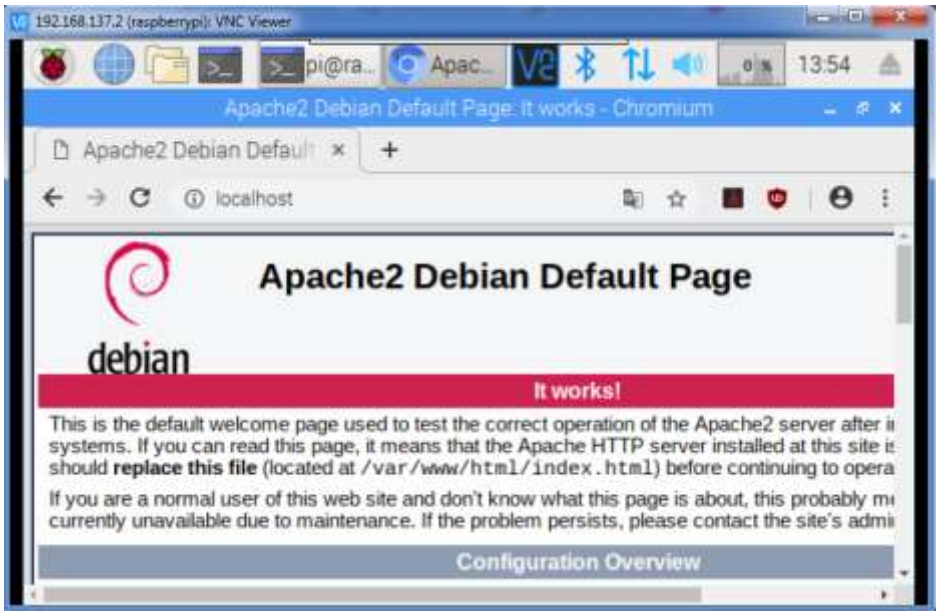

Figura 4.4 Página por defecto del servidor Apache en Debian.

Instalación de PHP

PHP es un lenguaje de código abierto de propósito general para el lado servidor muy extendido particularmente en el desarrollo web, que se caracteriza por la capacidad de ser incrustado dentro de código HTML. El interés de introducir de PHP en la Raspberry reside en la posibilidad de crear contenido dinámico.

Para proceder a la instalación de PHP en la Raspberry se han de ejecutar los siguientes comandos:

```
$ sudo apt-get install php
$ sudo apt-get install libapache2-mod-php libapache2-mod-perl2 php php-cli php-common php-curl php-dev php-gd php-imap php-ldap  php-mysql php-odbc
$ sudo reboot
```

Si una vez que se haya reiniciado se desea comprobar la correcta instalación, se podrá realizar creando un fichero llamado info.php en la siguiente ruta /var/www/html (directorio raíz del servidor web) con el siguiente contenido:

```
<?php
        phpinfo();
?>
```

Cuando se direccione la página info.php el navegador deberá volcar la siguiente página.

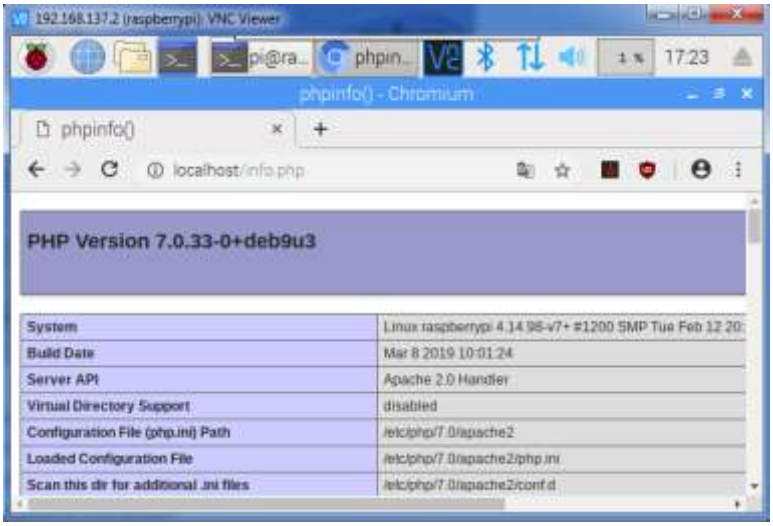

Figura 4.5 Información de PHP instalado en el sistema.

Configuración red del servidor

Para entender el siguiente apartado, se ha de explicar primeramente que la instalación de un DNS dinámico será para la implementación final del sistema, en la que clientes de fuera de la red privada, accederán al servidor desde la red pública para controlar los dispositivos electrónicos que se deseen.

Puesto que el sistema está diseñado para acceder desde cualquier equipo de la red, no es lo más inteligente el acceso al servidor mediante IP, ya que la IP de la parte pública es adjudicado dinámicamente, pudiendo cambiar en cualquier momento, lo que inutilizaría las comunicaciones de los clientes con el servidor. Para que esto no ocurra se usará un DNS, que tiene la función de asignar un dominio a IP, es decir, cuando los usuarios usen la web pondrán el nombre del dominio y no la IP, siendo DNS el encargado de traducir ese dominio a la IP que corresponda en ese momento, salvando así el sistema de posibles cambios.

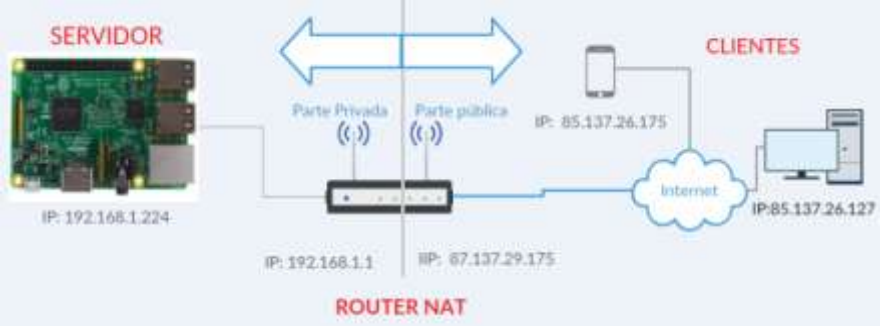

Figura 4.6 Ejemplo de esquema de red IP para la aplicación del proyecto.

Dado que el router hace NAT, cuando intenten acceder a la página web alojada al servidor, no lo hacen con la IP de la Raspberry, ya que esta pertenece a la red privada, lo hacen a través de la dirección IP de la parte pública del router, el cual se encarga de pasar la información que le llega desde la red pública a su destino en la red privada. Por este mismo motivo, la dirección IP a la que apunta el dominio tiene que ser la IP de la parte pública del router, ya que el cliente no sabe como encaminar un mensaje a una IP privada. En resumen, para internet la red privada no existe, la comunicación con ella y la salida a la red pública desde esta constan con la dirección IP de la parte pública del router como destino u origen respectivamente.

Volviendo a DNS, hay varias páginas que nos permiten crear un dominio para una IP dinámica. Para el proyecto se usará la web www.noip.com, que pese a tener funciones de pago, ofrece la posibilidad de mantener gratuitamente el dominio siempre y cuando lo renovemos cada 30 días. Además, no-ip avisa a sus usuarios cuando está a punto de pasar el tiempo por correo electrónico, por lo que no hay peligro de que el dominio caiga debido a un descuido del usuario. Para la creación del dominio se deberán realizar los siguientes pasos:

- El usuario se deberá primero registrar en no-ip seleccionando las opciones gratuitas.
- Pulsar en Managed DNS y seguidamente en la opción No-IP Hostnames de la pestaña desplegable Dynamic DNS.
- Introducción del Hostname . La dirección IP se introduce automáticamente con la que tenga el router en la parte pública actualmente.
- Creación del Hostname asegurando que se ha elegido la opción DNS Host (A), ya que este ofrece el servicio para IPv4.

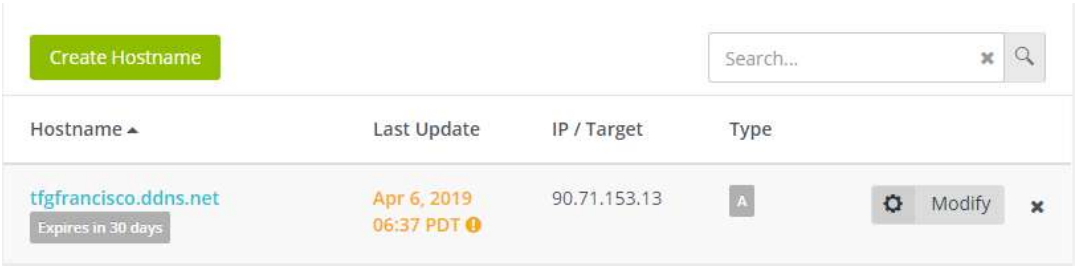

Figura 4.7 Dominio creado.

A continuación, para garantizar que el sitio web sea visible desde cualquier punto de la red se deberá trabajar en la Raspberry instalando ddclient, ya que es el encargado de notificarle a no-ip cuál es su actual IP pública, e introduciendo una serie de parámetros descritos a continuación:

- Instalación de ddclient con el siguiente comando.

```
$ sudo apt-get install ddclient
```

- En la pantalla de selección que aparece una vez acabada la instalación, se seleccionará www.noip.com como proveedor de servicio DNS.
- Introducción dyndns2 como protocolo de actualización de DNS dinámico, correo electrónico usado en el registro a no-ip como Username, contraseña en no-ip, wlan0 como interfaz de red, nombre completo del dominio, no ejecutar ddclient al establecer una conexión PPP, ejecutar el servidor ddns como demonio y ejecución de ddclient cada 300s.

Si se desea volver a reconfigurar el programa se podrá hacer mediante:

```
$ sudo dpkg-reconfigure ddclient
```

La configuración de ddclient final debería ser similar a la reflejada en la figura 4.8:

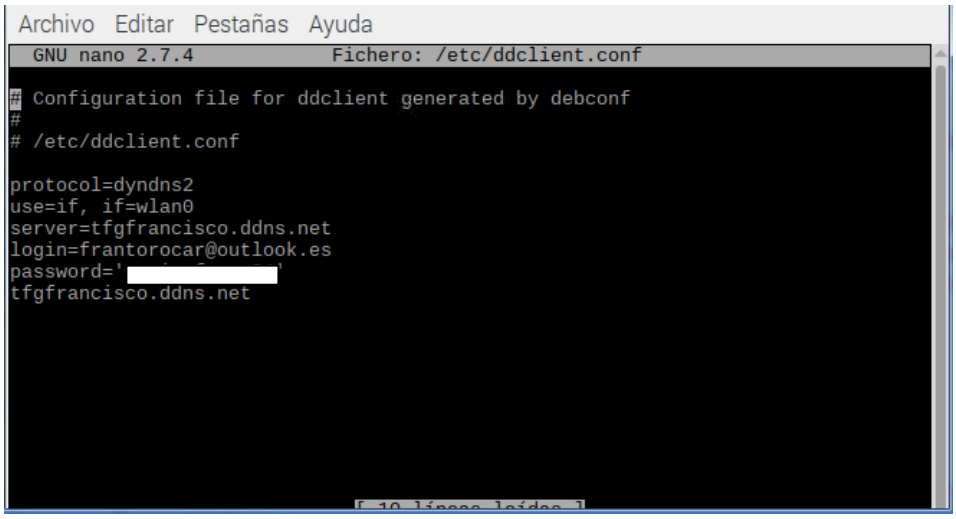

Figura 4.8 Configuración almacenada en el fichero ddclient.conf.

En este punto de la instalación se debe desconectar el cable de red y conectar la Raspberry Pi a la red inalámbrica. Para establecer una IP privada estática de red inalámbrica en la Raspberry Pi se debe acceder a la configuración de la red wlan0 en la Raspberry y configurar los parámetros como se visualiza en la figura 4.9.

Una vez hecho todo esto se puede abrir el puerto WEB (tcp/80) y DNS (tcp-udp/53) del Router para que los paquetes procedentes de la red con origen en el puerto 80 o 53 que llegan al router sean entregados al puerto 80 o 53 de la IP privada wlan0 de la Raspberry. Para ello se abre la configuración del router>avanzada>reenvío y se configura como se describe en la figura 4.10.

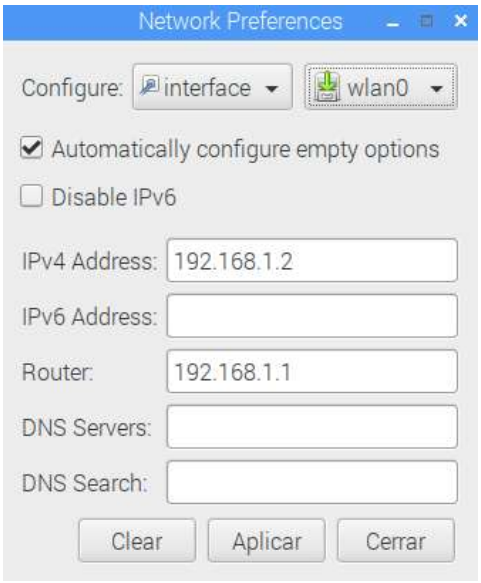

Figura 4.9 IP de la wlan0 de Raspberry Pi.

Finalmente se ha de reiniciar la Raspberry y el servidor Apache.

```
$ sudo reboot
$ sudo service apache2 restart
```

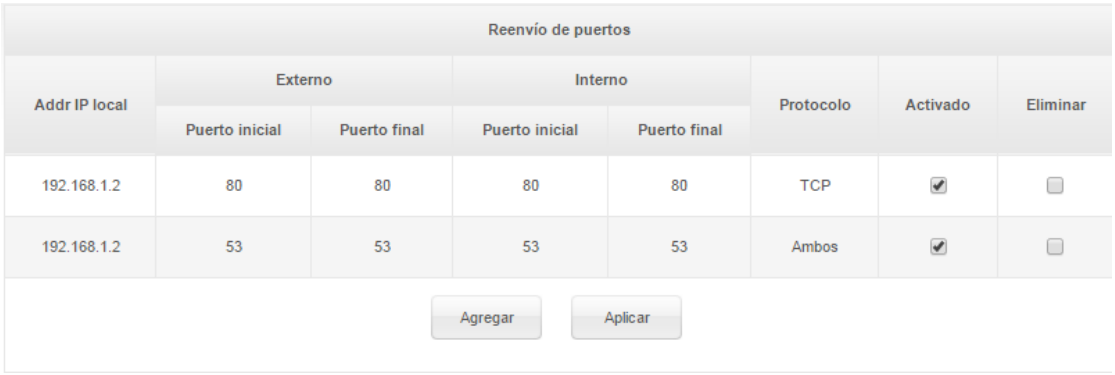

Figura 4.10 Apertura del puerto 80 y 53 al servidor.

Como punto final a toda la instalación y configuración, al direccionar el dominio (con el cable de red desconectado) desde fuera de la red local debe a aparecer la página por defecto del servidor.

Instalación del certificado SSL

La instalación de un certificado SSL necesita la existencia de un dominio. Para crear el certificado gratuito se va a recurrir a un certificado de Let's Encrypt. Para su instalación se ha de seguir los siguientes pasos:

```
$ sudo apt update
$ sudo apt upgrade
$ sudo rpi-update
$ sudo reboot
$ sudo apt-get install python-certbot-apache certbot
$ sudo nano /etc/apache2/sites-available/tfgfrancisco.ddns.net.conf
```

En el archivo abierto se ha de introducir la siguiente línea:

```
ServerName tfgfrancisco.ddns.net;
```

Ahora para obtener el certificado se introducirá:

```
$ sudo certbot --apache -d tfgfrancisco.ddns.net
```

Para la instalación del certificado se insta a la introducción del correo electrónico, el dominio, asentimiento de términos legales y si se quiere redirigir el tráfico de HTTP a HTTPS, lo cual es deseable.

Por último se debe reiniciar el servidor Apache y abrir el puerto SSL (443) del router. Cuando se introduzca la página web esta ya debería aparecer cifrada y como HTTPS.

5 PROGRAMACIÓN DEL ARDUINO

Para el control de equipos y la adquisición de datos en los equipos finales se empleará, como se ha descrito en otros capítulos, un microcontrolador Arduino Nano. Durante este capítulo se verá todas las librerías empleadas para el funcionamiento, así como las diferentes partes en las que se encuentra dividida el programa.

5.1. Librerías

En este punto se describirán las librerías que se han tomado de la comunidad de Arduino, así como el propósito por el que han sido incluidas y una breve descripción de su funcionamiento. A continuación se muestran las líneas de código que incluyen las librearías usadas.

```
#include <RCSwitch.h>
#include <Servo.h>
#include <DHT.h>
```

5.1.1 RCSwitch.h

Es una librería que permite el envío y recepción de señales radio de 315/433MHz a una tasa binaria aproximada de unos 10Kpbs con los módulos descritos en la sección 1.1.1. Básicamente se encarga de enviar una señal modulada en ASK.

En la mayoría de equipos de 433MHz del mercado estos módulos vienen acompañados de un chip, normalmente el SC5362, el HX2262 o el PT2262, encargado de generar un flujo de bits específico, sin embargo para el proyecto este flujo será proporcionado por la salida de un pin digital del Arduino, el cual constituye una herramienta mucho más flexible y potente para la construcción del mensaje a enviar.

ASK

La modulación por desplazamiento de amplitud (Amplitude-shift keying) consiste en variar la amplitud de la onda portadora en función de la palabra digital a enviar, manteniendo la frecuencia y la fase constante.

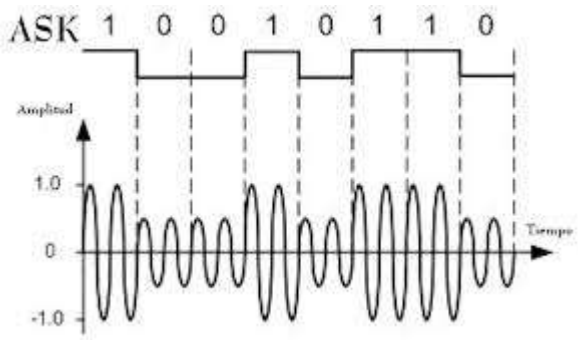

Figura 5.1 Modulación ASK.

Debido a la linealidad de esta modulación es muy sensible al ruido atmosférico y a las distorsiones, por eso es necesario una gran amplitud de banda que aunque consume más energía que otras modulaciones, lo

compensa con la sencillez de su modulación y de los equipos necesarios para enviarla o recibirla.

Funciones

La librería RCSwitch.h incluye las siguientes funciones:

- mySwitch.enableReceive(0): habilita la Interrupción 0 dada en el pin D2 para la recepción.
- mySwitch.enableTransmit(3): habilita la transmisión en el pin D3.
- mySwitch.available(): si ha llegado alguna señal la interrupción levanta una bandera.
- mySwitch.getReceivedValue(): devuelve el valor recibido.
- mySwitch.send(x,y): envía el valor x en y bits.

5.1.2 Servo.h

La librería servo.h se encarga de crear la señal PWM usada por el servo para mover su rotor a la posición deseada.

El procedimiento consiste en crear una clase servo ligada a un pin mediante la función attach() y mover el servo a la posición requerida con el comando write().

PWM

Es una señal eléctrica que repite una serie de pulsos con un ancho determinado. Dentro de un ciclo, el tiempo que la señal se encuentra a nivel alto determina el valor modulado, siendo 0 si la señal está a nivel alto el mínimo tiempo establecido y 180 si está el máximo el tiempo a 1. El valor mínimo y máximo se controla con la función attach(), siendo por defecto 544 y 2400 microsegundos respectivamente.

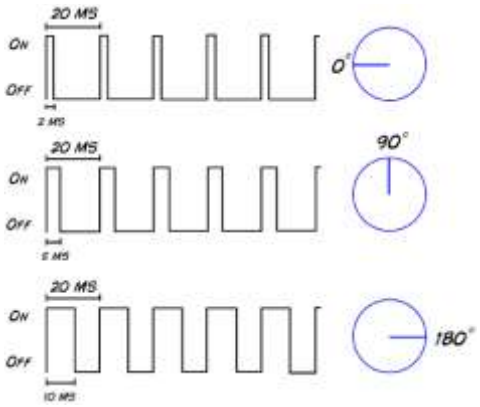

Figura 5.2 Ejemplo de modulación PWM.

Funciones

La librería incorpora las siguientes funciones:

- attach(pin, min, max): establece el pin que se encarga de controlar el ángulo del servo, siendo min el tiempo que debe estar a 1 para que el ángulo sea 0º y max el tiempo a 1 para que sea 180º.

- write(angle): mueve el servo al ángulo deseado.
- read(): devuelve el valor del ángulo correspondiente al valor pasado al comando write en la última llamada.
- attached(): comprueba si una variable de tipo servo se encuentra ligada a algún pin.

5.1.3 DHT.h

Es una librería que permite controlar los dispositivos DHT11, DHT12, DHT21 y DHT22 para la lectura en tiempo real de humedad y temperatura mediante las funciones readHumidity() y readTemperature() respectivamente.

5.2. Código

El propósito de esta sección es la explicación del código paso a paso, siendo el objetivo principal explicar la lógica del programa, atendiendo a las estructuras de control.

5.2.1 Definición de variables y constants

Justo al principio del programa se lleva a cabo la declaración de variables y constantes. Las variables definidas en este punto del programa tienen un carácter global, es decir, son accesibles y modificables desde cualquier punto de este.

```
#include <Servo.h>
#include <DHT.h>                                    //libreria DHT para el
sensor de humedad/temperatura

#define DHTPIN A0                                   // Definición del pin
digital donde se conecta el sensor de humedad y temperatura ambiente
#define DHTTYPE DHT11                               // Dependiendo del tipo de
sensor

#define sensorhumedad A3                            //pin para el sensor de
humedad del suelo

DHT dht(DHTPIN, DHTTYPE);                           //Inicialización el sensor
DHT11

const int rele[] = {4,5,6,7,8};                     //Pines que van a la entrada
de los reles
int valor;                                          //para almacenar el valor
que envía el tx
float h;                                            //humedad
float t;                                            //temperatura

int envio;                                          //almacena el valor de la
variable que se tx desde arduino

Servo servoMotor[5];

RCSwitch mySwitch = RCSwitch();                     //Libreia para la
emision/recepcion de 433MHz

unsigned long inicioMillis[]={0,0,0,0,0};           //variables para controlar
la temporización del rele sin ocupar el micro en la espera
unsigned long actualMillis[]={0,0,0,0,0};
unsigned long tiempo[]={0,0,0,0,0};
```

```
int equipoint=0;                                    //variable global con el
numero entero decimal del equipo que se quiere controlar
int estadorele[]={0,0,0,0,0};
```

La tabla de enteros llamada estadorele almacena en cada posición un 0 o un 1 dependiendo de si el relé correspondiente está activo o apagado. No obstante la función que asigna los valores a esta tabla será explicada más adelante.

5.2.2 Setup

La function setup() es llamada cuando empieza a ejecutarse el sketch. Es usada para inicializar variables, establecer los pines digitales/analógicos y arrancar librerías entre otras funciones. Solo se ejecuta una vez, bien sea al encender el Arduino o al resetearlo.

```
void setup()
{
  Serial.begin(9600);
  mySwitch.enableReceive(0);     // Interrupcion 0->pin Dig2
  mySwitch.enableTransmit(3);    // El pin  a usar para tx es el Dig3
  pinMode(rele[0], OUTPUT);
  pinMode(rele[1], OUTPUT);
  pinMode(rele[2], OUTPUT);
  pinMode(rele[3], OUTPUT);
  pinMode(rele[4], OUTPUT);
  pinMode(sensorhumedad, INPUT);
  servoMotor[0].attach(12);
  servoMotor[1].attach(11);
  servoMotor[2].attach(10);
  servoMotor[3].attach(9);
  servoMotor[4].attach(13);
  servoMotor[0].write(0);
  servoMotor[1].write(0);
  servoMotor[2].write(0);
  servoMotor[3].write(0);
  servoMotor[4].write(0);

}
```

Serial.begin(9600) permite la comunicación serie a 9600 baudios entre el Arduino y el ordenador, lo cual es de gran utilidad cuando se desea ver el valor de ciertas variables en una pantalla, siendo por tanto, una importante herramienta de depurado de código.

Las demás líneas de código tienen el propósito de establecer las señales de control de los relés como salidas, y de inicializar los servos atendiendo a las funciones de la librería servo.h explicada anteriormente.

5.2.3 Loop

Tras la definición del setup, el loop se ejecuta reiteradamente en bucle ejecutando las funciones requeridas en el programa como por ejemplo poner a nivel alto o bajo pines digitales, leer entradas analógicas, cambiar valores de variables…

Tradicionalmente el loop tiene las funciones de director de orquesta, es decir, lleva el guión principal del programa, pero este puede llamar a funciones auxiliares que le ayudan en la ejecución, haciendo que el código está más estructurado, sea más legible y sea mucho más fácil de depurar. Por este motivo, primero se expondrá el loop y seguidamente las funciones a las que llama.

El loop se articula en dos secciones, un if y un else if como se muestra en la figura 5.3, las cuales se van a

diferenciar porque tienen propósitos distintos.

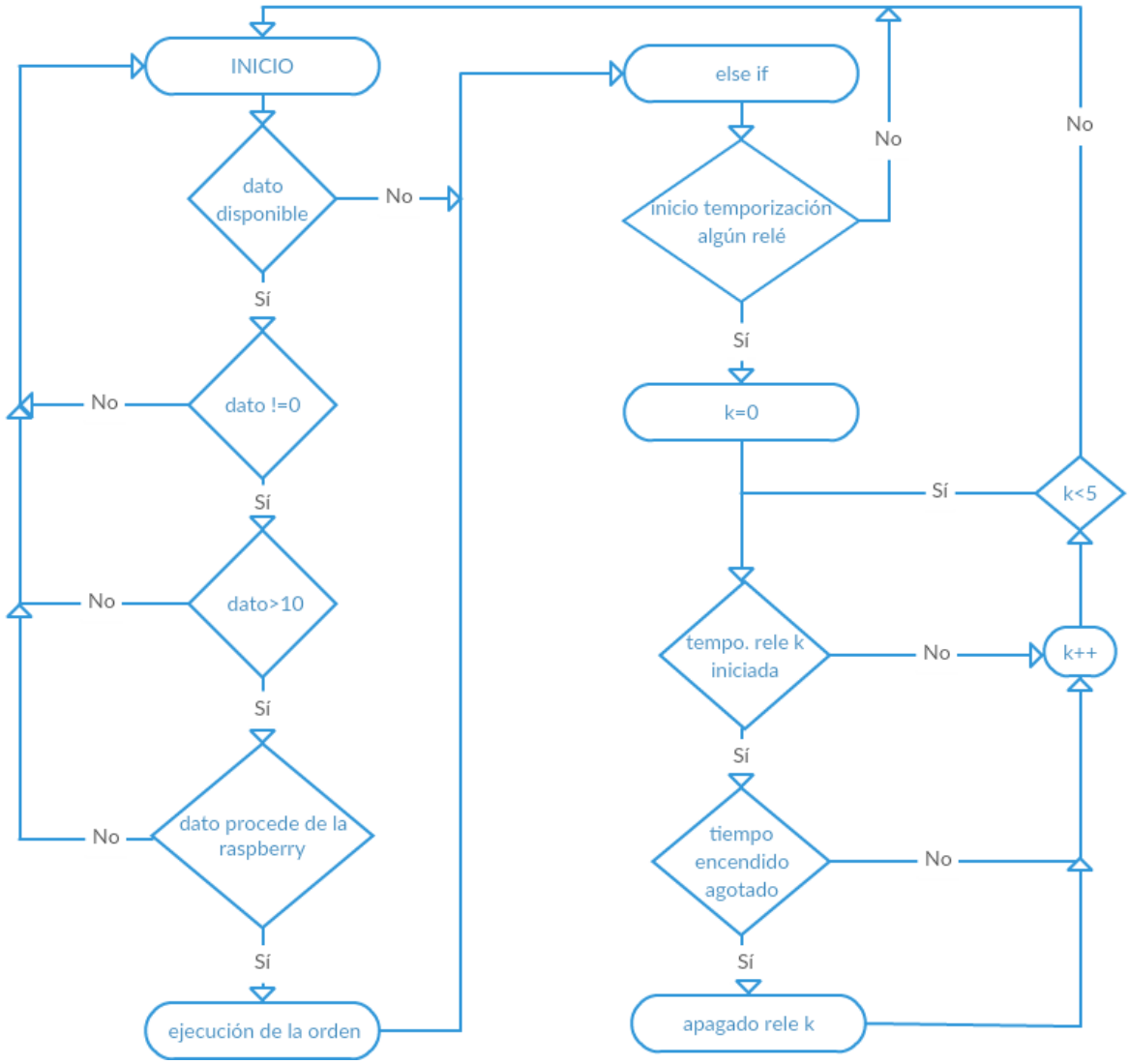

Figura 5.3 Diagrama de flujo del loop.

IF

```
inicio:;                                          //etiqueta
    if (mySwitch.available())
    {
    int value = mySwitch.getReceivedValue();
    if (value == 0)
    {
      Serial.print("Unknown encoding");
    }
```

```
    else
    {
      delay(500);                    //tiempo espera evitar recursividad
      Serial.print(valor);
      Serial.print("\n");
      if(value>10)                   //valor<10 es un espureo
      {
        valor=value;
      }
      else
        goto inicio;

      //Serial.print(valor);
      int bandera=ack(valor);        //comprueba procedencia mensaje
      if(bandera==0)
      {
        goto inicio;
      }
      String binario=calculabinario(valor);  //cadena binaria de valor
      String equip = binario.substring(11,15);
      String dats= binario.substring(15,binario.length());
      controlequipos(equip,dats);            //Funcion control equipos

    }
    mySwitch.resetAvailable();
    Serial.println("  ");
  }
```

El funcionamiento de este primer if es sencillo, cuando llega algún dato al receptor, y este entra dentro del rango de valores válidos, se llama a la función ack, que es la encargada de comprobar que la recepción procede del emisor de este sistema. Aunque se verá más tarde su funcionamiento, esta básicamente levanta una bandera que toma el valor 1 si todo ha ido bien, por lo que sigue la ejecución normal del programa, o no ha recibido asentimiento y por tanto desecha el valor volviendo al principio. En caso de que se reciba el asentimiento, se calcula la cadena binaria correspondiente al valor, de la cual se sustrae el número de equipo a controlar y los datos alusivos a dicho equipo. Completado todo lo anterior, se llama a la función controlequipos() que ejecuta las acciones deseadas.

El tiempo de espera del principio con delay() se usa porque el emisor envía varias veces la misma emisión una tras otra, lo que provoca que al recibir la primera emisión y ejecutar el código del ack, se interfieran las comunicaciones de ambos extremos. Por este motivo, tras recibir el primer mensaje, se esperan 500 microsegundos, que es el tiempo en que se estima que el emisor ya habrá emitido todo el mensaje correspondiente a la misma transmisión.

ELSE IF

```
  //gestiona el temporizado del rele iniciado desde el apartado rele
  else if (inicioMillis[0]!=0 || inicioMillis[1]!=0 || inicioMillis[2]!=0 ||
inicioMillis[3]!=0 || inicioMillis[4]!=0)
  {
    for (int k=0;k<=4;k++)
    {
      if(inicioMillis[k]!=0)
      {
```

```
        //nºde milisegundos desde que se inicio la funcion millis
        actualMillis[k] = millis();
        //1ms=(1/1000)*1s=(1/1000)*(1/60)*1min
        if ((actualMillis[k] - inicioMillis[k])/60/1000 >= tiempo[k])
        {
          inicioMillis[k] = 0;
          actualMillis[k] = 0;
          digitalWrite(rele[k], LOW);
        }
      }
    }
  }
```

Este segundo bloque tiene como propósito gestionar la temporización del encendido de relés, cuyo inicio se marca desde la sección correspondiente en la función controlequipos(). La gestión de este tiempo se lleva a cabo mediante la función millis(), la cual devuelve el valor en milisengundos que han transcurrido desde la primera llamada, permitiendo además temporizar el sistema sin la necesidad de parar el transcurso de este. Debido a que la función millis() continua paralelamente al programa desde su primera ejecución, el tiempo transcurrido en un determinado relé calcularse realizando la resta entre el tiempo actual y el tiempo en el que comenzó a temporizarse ese relé.

La función millis() desborda las variables de control (unsigned long int de 32 bits) en aproximadamente 50 días, como se muestra en la ecuación 5.1. Como no es posible poner a 0 la función millis() una vez que se inicia, es aconsejable aplicar un reset al Arduino Nano antes del tiempo estimado.

$$\text{nºdías} = \frac{2^{32} - 1}{1000 * 3600 * 24} = 49{,}71 \text{ días} \qquad (5.1)$$

5.2.4 ack

```
int ack (int recep)
{
  int ack;
  unsigned long ackMillis;
  unsigned long iniackMillis=millis(); //variable tiempo de espera del ack
  int flag=0;
  delay(300);
  mySwitch.send(recep,24);
  delay(300);
  ackMillis=millis();
  while((ackMillis-iniackMillis)<8000)        //espera de 8s
  {
   if (mySwitch.available())
    {
      ack= mySwitch.getReceivedValue();
      if (ack == 0)
      {
        Serial.print("Unknown encoding");
        flag=0;
        return flag;
      }
```

```
      else
      {
        if(ack==4773)
        {
          flag=1;
        }
        return flag;
      }
    }
  ackMillis=millis();
  }
  Serial.print("\n");
  return flag;
}
```

Esta función envía un mensaje, el cual es recibido por la Raspberry si y solo si esta ha sido la transmisora del mensaje recibido en el loop de Arduino. Esto es así, porque la Raspberry solo se pone a la escucha de un mensaje (ack) si ha transmitido algo primero, por lo tanto si no ha emitido nada no recibirá nada.

Si transcurren los 8 segundos de espera sin respuesta o se recibe una respuesta que no es la deseada, Arduino desecha el mensaje y sale de la función.

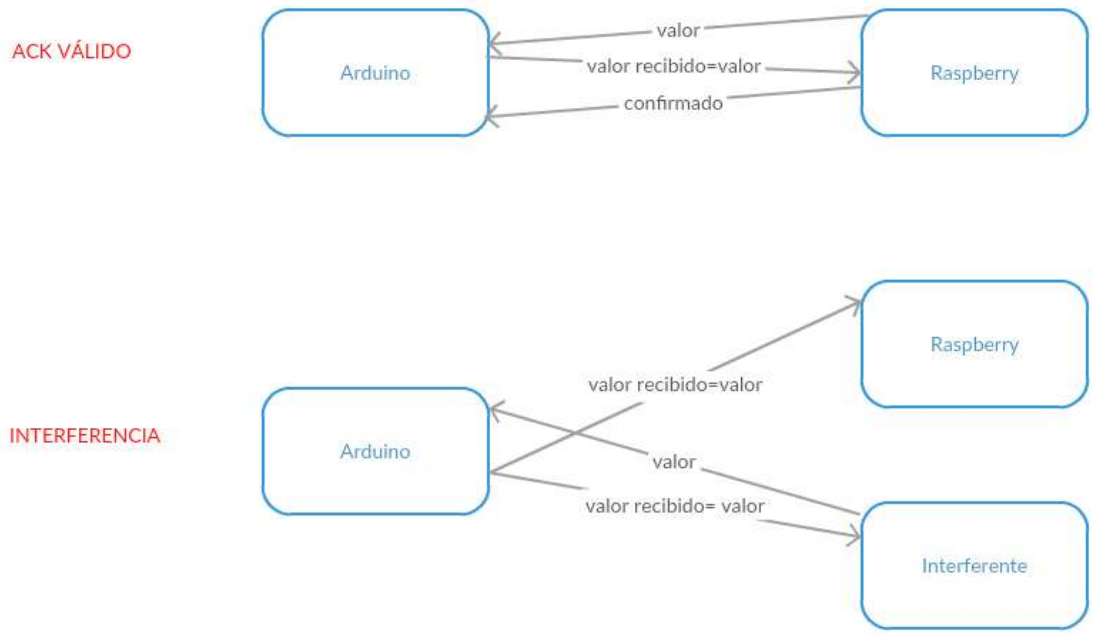

Figura 5.4 Función ack.

5.2.5 calculabinario

Para esta función se ha de tener en cuenta que el mensaje transmitido tiene la estructura mostrada en la figura 5.5.

Figura 5.5 Estructura del mensaje.

Los bits de ambos extremos son para devolver la línea a reposo, por lo que no forman parte del mensaje recibido. La parte sin uso se ha conservado por mantener la analogía con la longitud de mensaje de los dispositivos en el mercado, favoreciendo una futura adaptación para hacerlos interoperables.

```
String calculabinario (int value)
{
  //llega la cadena binaria que se envió en el tx pero con los
  //numeros binarios invertidos, es decir,
  //tx->11111111111000010111111111 se descarta el último y primer bit
  //tx-> 11111111110001011111111 y ahora se invierten los bits
  //rx-> 00000000001110100000000
  //la variable value es el valor binario rx pasado a decimal
  String binar = String(value, BIN); //se pasa el valor entero a una cadena
//de bits


  //Se tiene que tener siempre el mismo largo de cadena, para poder truncar
correctamente
  //y como lo primero que  llega son 0's a la izquierda estos se pierdan
  //por ejemplo tx(00011)-->valor-rx=3-->rx(11)
  //por eso se añanden a mano
  binar= "0000000000"+binar;

  binar.replace("1", "3");          //se invierte los valores para que
correspondan con los
  binar.replace("0", "1");          //que se emiten en la fuente. Se puesto
un 3
  binar.replace("3", "0");          //para no machacar el valor más tarde

  return binar;
}
```

5.2.6 controlequipos

Los equipos se distinguen entre: reles, servos, sensor de temperatura y humedad, y sensor de humedad del suelo, por lo que cada equipo tendrá su propia sección de control, y en el caso de los relés una más que averigua su estado.

Relés

```
//para controlar los rele
  if(equipo=="0000" || equipo=="0001" ||equipo=="0010" || equipo=="0011" ||
equipo=="0100" )
  {
    equipoint=bintodec(equipo);
//se averigua el numero entero decimal del equipo a controlar
    Serial.print(equipoint);
    if(datos=="11111111")
//si no se le pasa ningun dato en el formulario, encendido indefinido
      digitalWrite(rele[equipoint], HIGH);
    if(datos== "00000000")
//apagar rele
      digitalWrite(rele[equipoint], LOW);
    if(datos!="11111111" && datos!="00000000")
    {
      digitalWrite(rele[equipoint], HIGH);
      inicioMillis[equipoint] = millis();
//inicio de la cuenta de tiempo
      tiempo[equipoint]=bintodec(datos);
    }

  }
```

Se puede encender el relé de forma permanente, durante un tiempo determinado o apagarlo. Si se desea encender durante un tiempo determinado, se inicia la cuenta del tiempo para la temporización en el loop.

Servos

Puesto que se dispone 8 bits para modular el ángulo (0-255) y que el ángulo está comprendido entre 0 y 180º, el ángulo se modula mediante una regla de tres en el emisor (5.2), obteniéndose el valor deseado en el receptor deshaciendo esta operación.

$$\text{datos recibidos} = \frac{\text{ángulo} \times 255}{180} \qquad (5.2)$$

```
//para controlar el servo
  if(equipo=="0101" || equipo=="0110" ||equipo=="0111" || equipo=="1000" ||
equipo=="1001" )
  {
    int dato=bintodec(datos);          //para pasar la cadena binaria a
//decimal entero
    unsigned int angulo=dato*180;
    int equip=bintodec(equipo);
    angulo=angulo/255;                 //hay 8bits para modular el ángulo por
//lo que puede tener valores de 0 a 255
    Serial.print("\\");
    Serial.print(equip);
    Serial.print("\\");
    servoMotor[equip-5].write(angulo);
  }
```

Sensor de Humedad y Temperatura

```
  //para controlar el sensor de humedad y temperatura
  if(equipo=="1010")
  {
    dht.begin();                    // Comienza el sensor DHT
    h = dht.readHumidity();         // Lectura de la humedad
    t = dht.readTemperature();      // Lectura de  la temperatura en grados
centígrados (por defecto)


    if (isnan(h) || isnan(t))
    {
      return;
    }

    else
    {
      if(datos=="10101010")
      {
        mySwitch.send(h,24);        //envío del valor medido a la raspberry
      }
      if(datos=="01010101")
      {
         mySwitch.send(t,24);
      }

    //Serial.print("Humedad: ");
    //Serial.print(h);
    //Serial.print(" %\t");
    //Serial.print("Temperatura: ");
    //Serial.print(t);
    //Serial.print(" *C ");
    //Serial.print("\n");
  }
  }
```

Sensor de Humedad del suelo

```
    //para controlar el sensor de humedad y temperatura
  if(equipo=="1011")
  {
      //Se hace la lectura analogica del pin A0 (sensor) y se pasa por la
//funcion
      //map() para ajustar los valores leidos a los porcentajes que se
//quieren utilizar
      //lectura 0-->100(%)humedad
      //     1023-->0(%)
      int valorHumedad = map(analogRead(sensorhumedad), 0, 1023, 100, 0);
      mySwitch.send(valorHumedad,24);
      //Serial.print("Humedad: ");
      //Serial.print(valorHumedad);
      //Serial.println("%");
  }
```

Estado de los relés

Para averiguar el estado de los relés se recurre a la función digitalRead(pin), la cual devuelve un 1 si ese pin digital está a nivel alto y un 0 si está a nivel bajo. Para pasar el estado de todos los relés en un solo mensaje se

usa la fórmula 5.3, que sigue la analogía de pasar un número binario a uno decimal.

```
  //para ver que pines de los reles hay activos
  if(equipo=="1100")
  {
    estadorele[0]=digitalRead(4);
    estadorele[1]=digitalRead(5);
    estadorele[2]=digitalRead(6);
    estadorele[3]=digitalRead(7);
    estadorele[4]=digitalRead(8);
    envio=pow(2,4)*estadorele[0]+
pow(2,3)*estadorele[1]+pow(2,2)*estadorele[2]+ pow(2,1)*estadorele[3]+
estadorele[4]+1;
 //Se le sumam 1 porque si no hay ninguno activo no se podria mandar el valor
0-->se lo resta en el destino
    mySwitch.send(envio,24);
  }
```

$$envio = estado0 \times 2^4 + estado1 \times 2^3 + estado2 \times 2^2 + estado3 \times 2^1 + estado4 + 1 \qquad (5.3)$$

6 PROGRAMACIÓN DE LA RASPBERRY PI

La Raspberry Pi como bien se ha explicado antes, es tanto la encargada de albergar el servidor web en el que se aloja la página web para el control del sistema, como de transmitir por RF las peticiones realizadas en la página web al Arduino. También recibe datos de sensores y asentimientos.

En esta sección se describirá primero brevemente los lenguajes de programación empleados seguido de una descripción de los códigos empleados.

6.1. Lenguajes de programación

Dado que cada lenguaje de programación tiene su funcionalidad, se hace imprescindible una breve explicación de todos los lenguajes empleados.

6.1.1 HTML

Es un lenguaje usado para la creación de páginas web. Se usa para estructurar el contenido de una página web mediante etiquetas, que indican el tipo de contenido que encierra.

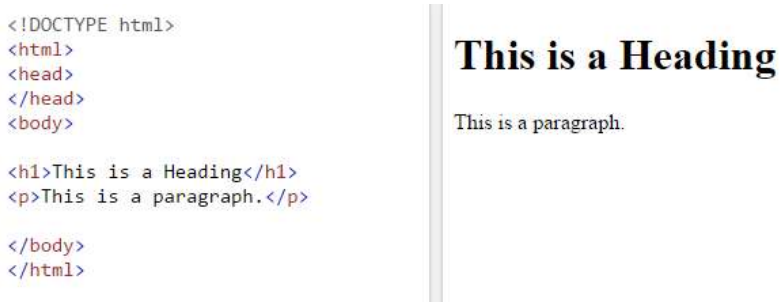

Figura 6.1 Código HTML a la izquierda y representación del código en el navegador a la derecha.

6.1.2 css

Es usado para definir el diseño visual de un documento web escrito en HTML.

Figura 6.2 Ejemplo de estilo CSS aplicado a código HTML.

6.1.3 PHP

Es un lenguaje de código abierto especializado para el desarrollo web que puede ser incrustado dentro de HTML.

Es un código ejecutado en el lado servidor. El cliente recibe el resultado de ejecutar el programa, pero no recibe el código empleado, lo cual mejora la privacidad del lado servidor, protegiendo contraseñas y demás contenido sensible.

En el servidor PHP será usado principalmente para ejecutar la orden que transmite la señal RF en base a un formulario web y para la gestión de contraseñas.

6.1.4 Javascript

Se usa para programar el lado cliente, permitiendo mejoras de interfaz usuario y la posibilidad de crear web dinámicas.

En el programa es usado para generar una tabla dinámica que indica los relés activos en base a la señal RF recibida por la Raspberry Pi.

6.1.5 Python

Python es un lenguaje interpretado multiparadigma, ya que soporta orientación a objetos, programación imperativa y funcional, enfocado a favorecer la legibilidad del código.

Dado que posee una licencia de código abierto compatible con GNU, es totalmente operable en Raspberry Pi, lo cual lo convierte en una potente herramienta para transmitir y recibir transmisiones RF a través de los GPIO.

6.1.6 C++

Este lenguaje, además de para programación genérico o estructurada, permite extender el lenguaje c con la posibilidad de manipular objetos.

C++ será usado para gestionar la recepción de mensajes RF del sistema.

6.2. 433Utils

Se compone de una serie de archivos que permiten recibir emisiones de 433MHz. Para su uso se ha de seguir una serie de pasos que se van a exponer en esta sección de forma ordenada.

Primero se debe instalar WiringPi, ya que permite el control de los GPIO de la Raspberry Pi y por tanto el uso de estos para la recepción y emisión, mediante los siguientes comandos:

```
$ cd ~/
$ git clone git://git.drogon.net/wiringPi
$ cd wiringPi/
$ ./build
```

Una vez acabado el anterior paso ya se puede proceder a instalar 433Utils, el cual debe ser instalado en la carpeta html, ya que se necesita tener acceso desde el servidor web.

```
$ cd /var/www/html
$ sudo git clone -- recursive git://github.com/ninjablocks/433Utils.git
$ cd 433Utils/RPi_utils
$ sudo make
```

Puesto que el ejecutable del archivo c++ se queda a la escucha todo el tiempo, se va a modificar para que salga tras el primer mensaje recibido o al pasar dos segundos sin recibir nada desde su ejecución. Así quedaría el fragmento de código modificado:

```
    while(bandera==0) {
      t = (double) clock();        /* tiempo que ha pasado */
      t = t / CLOCKS_PER_SEC;      /*pasa la medida a segundos*/
      if(t>2)                      //si no llega nada en 2s
      {
        return 0;
      }
      if (mySwitch.available()) {
        bandera=1; //bandera que nos indica que hemos entrado al bucle y que
//no queremos entrar más
        int value = mySwitch.getReceivedValue();

        if (value == 0) {
          printf("Unknown encoding\n");
        } else {

          printf("%i\n", mySwitch.getReceivedValue() );
          salvar=value;
          mySwitch.resetAvailable();
        }

        fflush(stdout);
      }
      usleep(100);

  }
```

La parte añadida está formada por una aplicación de la librería time.h la cual sirve para contabilizar el tiempo que ha transcurrido desde que se ejecutó el programa, y por una variable denominada bandera que indica que ya se ha leído un mensaje y por tanto se ha de salir del programa. En resumen, el programa estará ejecutándose 2 segundos si no recibe nada o hasta que reciba un mensaje antes de dicho tiempo. En caso de que se haya recibido algo este valor será devuelto en la variable salvar, que toma el valor de la variable value si es válida.

C++ no es un código interpretado, por lo que no se debe olvidar compilar para crear el nuevo ejecutable.

```
$ sudo make
```

6.3. TransmitRF.py

Partiendo del archivo TransmitRF.py ([1]), el cual solo permite encender o apagar dispositivos, se ha realizado una serie de modificaciones que permiten componer un mensaje que sea capaz de identificar y controlar los dispositivos adecuadamente, tal y como se muestra en la figura 5.5.

La modificación básicamente compone el mensaje en base a unos parámetros pasados desde el formulario HTML y seguidamente envía el flujo binario al GPIO para su transmisión.

```
#La primera parte de eleccion,(que es la cadena a tx)
#es siempre igual.Hay que tx un 0 para que salga del
#reposo, que es tx 1's
eleccion = '111111111110'

#Los 4 siguientes bits identifican al dispositivo a #controlar

#Seccion para controlar el rele
if sys.argv[2] == "rele":
    releint = int(sys.argv[3])-1 #String->Entero

    #La funcion binario convierte un entero a cadena de
    #caracteres
    bin = binario(releint)

    #se debe llenar hasta 4 para que siempre se
    #tx la misma cantidad de bits. Esto luego nos facilitara
    #la vida a la hora de deshacer el cambio a entero en rx

    while len(bin) < 4:
        bin = '0'+bin

    eleccion= eleccion + bin    #Concatenacion
    #Hay que tx un 1 al final minimo para que transmisor
    #vuelva a reposo
    #si no se le pasan los minutos en el formulario
    #se entiende que esta encendido permanentemente
    if sys.argv[4] == 'encender' and sys.argv[5]=='':
        eleccion = eleccion + '111111111'
        print (eleccion)
    if sys.argv[4] == 'apagar':
        eleccion = eleccion + '000000001'
        print(eleccion)
    if sys.argv[4] == 'encender' and sys.argv[5]!='':
        #Hacer una regla de 3
        #Tiene 8 bits para codificar minutos
        minu = int(sys.argv[5])
        minustr= str(binario(minu))
        while len(minustr) < 8:
            minustr = '0' + minustr;
        #Hay que tx un 1 al final minimo para que transmisor
        #vuelva a reposo
        eleccion = eleccion + minustr + '1'

    print(eleccion)
#Seccion para controlar el microservo
if sys.argv[2] == "servo":
    servoint = int(sys.argv[3])+4    #String->int
                                     #5 prim son reles
    servo= binario(servoint)          #int->bin

    while len(servo) < 4:
```

```
        servo= '0' + servo

    eleccion= eleccion + servo    #Concatenacion

    #Hacer una regla de 3
    #Tiene 8 bits para codificar el angulo
    angulo = int(sys.argv[4])*255
    angulo= angulo /180
    angulostr= str(binario(angulo))

    while len(angulostr) < 8:
        angulostr = '0' + angulostr;
    eleccion = eleccion + angulostr
    print (eleccion)
    #Hay que tx un 1 al final minimo para que transmisor
    #vuelva a reposo
    eleccion = eleccion + '1'

#Seccion para controlar el sensor de humedad y temp.
if sys.argv[2] == "humtem":
    #se le envia al rx una senal que indica unicamente que
    #tiene que tomar medidas del sensor de temp y hum.
    #En la modulacion de este sensor correspondera con el
    #equipo 10
    if sys.argv[3] == "humedad":
        eleccion= '111111111110'+'1010' + '101010101'
    if sys.argv[3] == "temperatura":
        eleccion= '111111111110'+'1010' + '010101011'

if sys.argv[2] == "humsuelo":
    #se le manda al rx una senal que indica unicamente que
    #tiene que tomar medidas del sensor de hum. suelo
    #En nuestra modulacion este sensor correspondera con el
    #equipo 11
    eleccion= '111111111110'+'1011' + '101010101'

if sys.argv[2] =="compruebareles":
    #se le manda al rx una senal que indica unicamente que
    #tiene que ver que reles estan activos
    eleccion= '111111111110'+'1100' + '101010101'

if sys.argv[2] =="ack":
    #se le manda al rx una senal que indica unicamente que
    #tiene que ver que reles estan activos
    eleccion= '111111111110'+'1101' + '010110101'
    print(eleccion);
```

6.4. Programación web

En este punto se recogen los archivos asociados a la parte web, constituyendo por tanto la interface usuario para controlar los equipos. Son también los encargados de coordinar todas las comunicaciones RF y pasar los parámetros adecuados a la función TransmitRF.py.

6.4.1 login.php

Es el encargado de gestionar el acceso al sistema mediante una contraseña. En caso de que se introduzca incorrectamente, no se podrá acceder a ninguna de las funciones, pero si la contraseña es correcta, mantendrá la sesión abierta hasta que se apague el servidor o se pulse el botón desconectar, evitándose así que se tenga que introducir la contraseña reiteradas veces en los distintos redireccionamientos que se producen en el

sistema.

La contraseña correcta disponible en el servidor está encriptada con md5, el cual es un algoritmo de reducción criptográfico de 128 bits, con el objetivo de que si se accede a los archivos del sistema, se observe la contraseña encriptada y no la real. Esto es así porque md5 permite encriptar, pero no desencriptar.

Figura 6.3 login.php representado en el navegador.

Dado que la contraseña se encuentra encriptada en el servidor, se ha de encriptar la contraseña introducida en el formulario para la comparación. Se ha de recalcar que la encriptación de la contraseña encriptada no es igual a la contraseña original.

```
<?php
$password = "a58a14c5390aa81d14063ce53134c24a";
session_start();

if($_POST['password'])
{
    if(md5($_POST['password']) == $password)
    {
        $_SESSION['password'] = "alm";
        $_SESSION['name']="Fran";
    }
```

Las líneas de código anteriores muestran como se abriría una sesión web en caso de que la encriptación de la contraseña introducida en el formulario coincidiera con la contraseña encriptada correcta que dispone el servidor.

El siguiente else, el cual se ejecuta si se ha iniciado sesión, redirecciona a la página inicio.php que se explicará a continuación.

```
else
{
?>
<a href="<?php echo $_SERVER['PHP_SELF']; ?>?desconectar=si">Desconectar</a>
<?php
    header("Location: inicio.php");
}
```

6.4.2 inicio.php

Es esencialmente una página que muestra un select con las distintas funciones del sistema, y en base a la seleccionada se redireccionará a una página distinta. La siguiente sección de código está tomada a trozos y está incompleta, pero sirve para clarificar lo anterior.

```
<select name="gadget">
                    <option value="1">Relé  </option>
                    <option value="2">Servo </option>
                    <option value="3">Hum y Tem </option>
                    <option value="4">Humedad del suelo </option>
                    <option value="5">Compruebareles</option>
                    <option value="6">Control raspberry </option>
                    <option
                </select>

if($_POST['Desconectar'] && !$_POST['Enviar'])
  {
    session_destroy();
    header("Location: login.php");
  }
if ($_POST[gadget]==1 && !$_POST['Desconectar'] && $_POST['Enviar'])
  {
    //redireccionamos a la página que controla los rele
    header("Location: rele.php");
  }
```

6.4.3 rele.php

La página rele.php permite controlar hasta 5 relés. Dentro de las opciones que ofrece la página se encuentran:

- Apagar relés.
- Encender relés indefinidamente.
- Encender relés con temporizado de 1 a 255 minutos.

La cadena eleccion es donde se compone el mensaje en el archivo TransmitRF.py. El paso de mensajes en las funciones sigue la estructura anteriormente explicada en el capítulo 5 tal y como se describe en la figura 5.5.

```
if ($_POST[encender] && !$_POST['Desconectar'])
{
    $a- exec("sudo python /var/www/html/TransmitRF.py  'eleccion' 'rele'
'$_POST[relesel]' 'encender' '$_POST[minutos]'");
    echo $a;
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");
}
if ($_POST[apagar] && !$_POST['Desconectar'])
{
    $a- exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'rele'
'$_POST[relesel]' 'apagar'");
    echo $a;
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");
```

```
}
```

6.4.4 servo.php

Permite controlar hasta 5 servos estableciendo la posición deseada siempre que se encuentre entre los 0 y 180º.

6.4.5 humtem.php

Esta página a diferencia de las anteriores, necesita el envío de datos desde Arduino, ya que es este el encargado de leer el valor de temperatura y humedad que se vuelca en la pantalla.

6.4.6 humsuelo.php

Tiene un funcionamiento similar a humtem.php salvo que esta muestra el valor en pantalla de la humedad del suelo.

6.4.7 controlareles.php

Recibe desde Arduino el estado de los relés codificado según la fórmula 5.3 vista con anterioridad. Tras deshacer el cambio, se crea con javascript una tabla dinámica que saca por pantalla On u Off para cada relé dependiendo de si es un 1 o un 0 respectivamente.

```
if ($_POST[Comprobar] && !$_POST['Desconectar'])
{

    $a=exec("sudo python /var/www/html/TransmitRF.py  'eleccion'
'compruebareles'");
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");
    $mreles = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");

    $mreles=$mreles-1;

    if ($mreles<0):
    ?>
        <p align="center" class="error">Mensaje de error:Módulos 433MHz con
alimentación insuficiente o no está transmitiendo correctamente</p>
    <?php
    else:
    $binre=decbin($mreles);
    $cadena=(string)$binre;
    while (strlen($cadena) <= 4):
        $cadena= '0' . $cadena;
    endwhile;
?>
    <script type="text/javascript">
    var cadenas = "<?php echo $cadena;?>";

    document.getElementById("1").innerHTML =cadenas[0];
    document.getElementById("2").innerHTML =cadenas[1];
    document.getElementById("3").innerHTML =cadenas[2];
    document.getElementById("4").innerHTML =cadenas[3];
    document.getElementById("5").innerHTML =cadenas[4];
    var x=document.getElementsByClassName("abajo");
    var k=0;
```

```
    var dato;
    var str;                        //almacena dato como string
    var n=0;
    while (k<=4)
    {

        dato=x[k].innerHTML;     //para ver el contenido de la celda k de la
tabla
        //document.write(dato);
        n=k+1;
        str=n.toString();
        if (dato==1)
        {
            document.getElementById(str).innerHTML ="On";
        }
        if (dato==0)
        {
            document.getElementById(str).innerHTML ="Off";
        }
        k=k+1;

    }

    </script>
<?php
     endif;
```

6.4.8 controlrasp.php

Si se detecta alguna anomalía es posible reiniciar la Raspberry o reiniciar el servidor Apache con esta página.

```
if ($_POST['reiniciosistema'] && !$_POST['reinicioservidor'])
{

    exec("sudo reboot");

}
if ($_POST['reinicioservidor'] && !$_POST['reiniciosistema'])
{
    exec("sudo service apache2 restart");
}
```

7 CONCLUSIÓN

En la ejecución de este proyecto podría decirse que se han solventado las principales carencias visibles del montaje de partida. Todo ello ha sido solucionado con herramientas de software libre y elementos hardware de muy bajo coste y reutilizables en su mayoría.

Como respuesta a uno de los problemas críticos del anterior sistema, la ausencia de un protocolo de encriptación que escondiera la contraseña de terceros, se ha conseguido satisfactoriamente la instalación de un certificado SSL autofirmado que permite realizar una conexión cifrada empleando aplicaciones gratuitas. Como medida adicional se ha introducido en el código php del servidor la contraseña correcta encriptada con el protocolo md5, permitiéndose así que si un tercero accede a dicho código sea incapaz de descifrar la contraseña verdadera.

Con respecto al problema de la comunicación simplex, se ha conseguido un resultado muy satisfactorio, ya que con la incorporación de una pareja de emisor y receptor en cada lado del sistema ha conseguido que la comunicación pase a ser half-dúplex, permitiéndose así la adquisición de datos de los sensores del Arduino y el establecimiento de un protocolo de asentimiento que permite discernir cuando a el Arduino le ha llegado una comunicación que no pertenece al lado servidor.

La inflexibilidad del sistema ha sido otro punto a cubrir, ya que el protocolo de envío de los módulos de 433MHz enviaba un tren de bits de una longitud bastante grande, pero estos que apenas permitían distinguir entre un par de funciones. Con la nueva modulación implementada, se permite reconocer hasta un total de 16 equipos, disponiéndose de hasta 8 bits para la modulación de datos.

7.1. Futuras mejoras

Dado que el sistema es totalmente libre, se podría configurar su funcionamiento con total libertad, añadiendo o quitando el número de equipos que se pueden controlar. La única limitación de equipos a controlar podría venir dada por el número de pines disponibles en el microcontrolador, que podría ser solventado con el cambio a un microcontrolador mayor de la familia Arduino como el Arduino MEGA. Igualmente los datos enviados podrían ser reducidos o aumentados, siempre adecuándose al tipo de dispositivo que se quiera controlar.

8 CÓDIGOS COMPLETOS

En esta sección se recogerán todos los códigos empleados para la ejecución del programa.

8.1. Código de Arduino

```
#include <RCSwitch.h>
#include <Servo.h>
#include <DHT.h>                                  //libreria DHT para el
sensor de humedad/temperatura

#define DHTPIN A0                                 // Definición del pin
digital donde se conecta el sensor de humedad y temperatura ambiente
#define DHTTYPE DHT11                             // Dependiendo del tipo de
sensor

#define sensorhumedad A3                          //pin para el sensor de
humedad del suelo

DHT dht(DHTPIN, DHTTYPE);                         //Inicialización el sensor
DHT11

const int rele[] = {4,5,6,7,8};                   //Pines que van a la entrada
de los reles
int valor;                                        //para almacenar el valor
que envía el tx
float h;                                          //humedad
float t;                                          //temperatura

int envio;                                        //almacena el valor de la
variable que se tx desde arduino

Servo servoMotor[5];

RCSwitch mySwitch = RCSwitch();                   //Libreria para la
emision/recepcion de 433MHz

unsigned long inicioMillis[]={0,0,0,0,0};         //variables para controlar
la temporización del rele sin ocupar el micro en la espera
unsigned long actualMillis[]={0,0,0,0,0};
unsigned long tiempo[]={0,0,0,0,0};

int equipoint=0;                                  //variable global con el
numero entero decimal del equipo que se quiere controlar
int estadorele[]={0,0,0,0,0};

void setup()
{
  Serial.begin(9600);
  mySwitch.enableReceive(0);                      // Interrupcion 0->pin Dig2
  mySwitch.enableTransmit(3);                     // El pin que vamos a usar
para tx es el Dig3
  pinMode(rele[0], OUTPUT);
  pinMode(rele[1], OUTPUT);
  pinMode(rele[2], OUTPUT);
  pinMode(rele[3], OUTPUT);
  pinMode(rele[4], OUTPUT);
  pinMode(sensorhumedad, INPUT);
```

```
  servoMotor[0].attach(12);
  servoMotor[1].attach(11);
  servoMotor[2].attach(10);
  servoMotor[3].attach(9);
  servoMotor[4].attach(13);
  servoMotor[0].write(0);
  servoMotor[1].write(0);
  servoMotor[2].write(0);
  servoMotor[3].write(0);
  servoMotor[4].write(0);

}

void loop()
{
inicio:;
//etiqueta
    if (mySwitch.available())
    {
    int value = mySwitch.getReceivedValue();
    if (value == 0)
    {
      Serial.print("Unknown encoding");
    }

    else
    {
      delay(500);
//tiempo de espera para que el programa no vuelva a entrar multiples veces en
la misma emision
      Serial.print(valor);
      Serial.print("\n");
      if(value>10)
//aparecen espúreos normalmente de poco valor al final de la comunicación,
      {
//provocando que el valor correcto se machaque y no se obtenga el valor
deseado. Por tanto se desechan.
        valor=value;
//prevalece el valor de la vez anterior
      }
      else
        goto inicio;

      //Serial.print(valor);
      int bandera=ack(valor);
//comprueba si la informacion que le ha llegado procede de nuestro emisor
      if(bandera==0)
      {
        goto inicio;
      }
      String binario=calculabinario(valor);
//funcion que calcula la cadena binaria del valor entero recibido en la rx
      String equip = binario.substring(11,15);
//bit del 12 al 15 indican el equipo
      String dats= binario.substring(15,binario.length());
//bit del 16 al 23 indican los datos enviados
      controlequipos(equip,dats);
//Funcion que controla los reles, sensores añadidos al arduino en base a los
datos recibidos

    }
    mySwitch.resetAvailable();
```

```
    Serial.println("  ");
  }
  //gestiona el temporizado del relese que se inicia desde el apartado rele
  else if (inicioMillis[0]!=0 || inicioMillis[1]!=0 || inicioMillis[2]!=0 ||
inicioMillis[3]!=0 || inicioMillis[4]!=0)
  {
    for (int k=0;k<=4;k++)
    {
      if(inicioMillis[k]!=0)
      {
        actualMillis[k] = millis();
//n°de milisegundos desde que se inicio la funcion millis
        if ((actualMillis[k] - inicioMillis[k])/60/1000 >= tiempo[k])
//comprobamos si ha pasado el tiempo 1ms=(1/1000)*1s=(1/1000)*(1/60)*1min
        {
          inicioMillis[k] = 0;
          actualMillis[k] = 0;
          digitalWrite(rele[k], LOW);
        }
      }
    }
  }
}

int ack (int recep)
{
  int ack;
  unsigned long ackMillis;
  unsigned long iniackMillis=millis();
//variable para determinar el tiempo de espera del ack
  int flag=0;
//para luego determinar si continua con la ejecucion del programa normal o
desecha el dato recibido anteriormente
  delay(300);
  mySwitch.send(recep,24);
  delay(300);
  ackMillis=millis();
  while((ackMillis-iniackMillis)<8000)
//si no se recibe una respuesta en 8s se desecha el mensaje recibido
  {
   if (mySwitch.available())
    {
      ack= mySwitch.getReceivedValue();
      if (ack == 0)
      {
        Serial.print("Unknown encoding");
        flag=0;
        return flag;
      }

      else
      {
        if(ack==4773)
        {
          flag=1;
        }
        return flag;
      }
    }
  ackMillis=millis();
  }
  Serial.print("\n");
```

```
  return flag;
}

String calculabinario (int value)
{
  //llega la cadena binaria que se envió en el tx pero con los
  //numeros binarios invertidos, es decir,
  //tx->11111111111000010111111111 se descarta el último y primer bit puesto a
1 (reposo)->
  //tx-> 11111111110001011111111 y ahora se invierten los bits
  //rx-> 00000000001110100000000
  //la variable value es el valor binario rx pasado a decimal
  String binar = String(value, BIN); //se pasa el valor entero a una cadena
de bits


  //Se tiene que tener siempre el mismo largo de cadena, para poder truncar
correctamente
  //y como lo primero que  llega son 0's a la izquierda estos se pierdan
  //por ejemplo tx(00011)-->valor-rx=3-->rx(11)
  //por eso se añanden a mano
  binar= "0000000000"+binar;

  binar.replace("1", "3");          //se invierte los valores para que
correspondan con los
  binar.replace("0", "1");          //que se emiten en la fuente. Se puesto
un 3
  binar.replace("3", "0");          //para no machacar el valor más tarde

  return binar;
}

void controlequipos(String equipo, String datos)
{
//para controlar los rele
  if(equipo=="0000" || equipo=="0001" ||equipo=="0010" || equipo=="0011" ||
equipo=="0100" )
  {
    equipoint=bintodec(equipo);
//se averigua el  a controlar
    Serial.print(equipoint);
    if(datos=="11111111")
//si no se le pasa ningun dato en el formulario, encendido indefinido
      digitalWrite(rele[equipoint], HIGH);
    if(datos== "00000000")
//apagar rele
      digitalWrite(rele[equipoint], LOW);
    if(datos!="11111111" && datos!="00000000")
    {
      digitalWrite(rele[equipoint], HIGH);
      inicioMillis[equipoint] = millis();
//inicio de la cuenta de tiempo
      tiempo[equipoint]=bintodec(datos);
    }

  }

  //para controlar el servo
  if(equipo=="0101" || equipo=="0110" ||equipo=="0111" || equipo=="1000" ||
equipo=="1001" )
  {
```

```
    int dato=bintodec(datos);
//para pasar la cadena binaria a decimal entero
    unsigned int angulo=dato*180;
    int equip=bintodec(equipo);
    angulo=angulo/255;
//hay 8bits para modular el ángulo por lo que puede tener valores de 0 a 255
    Serial.print("\\");
    Serial.print(equip);
    Serial.print("\\");
    servoMotor[equip-5].write(angulo);
  }

  //para controlar el sensor de humedad y temperatura
  if(equipo=="1010")
  {
    dht.begin();
// Comenzamos el sensor DHT
    h = dht.readHumidity();
// Leemos la humedad
    t = dht.readTemperature();
// Leemos la temperatura en grados centígrados (por defecto)


    if (isnan(h) || isnan(t))
    {
      return;
    }

    else
    {
      if(datos=="10101010")
      {
        mySwitch.send(h,24);
//envío del valor medido por el sensor al servidor para poder ser mostrado en
la web
      }
      if(datos=="01010101")
      {
         mySwitch.send(t,24);
      }

    //Serial.print("Humedad: ");
    //Serial.print(h);
    //Serial.print(" %\t");
    //Serial.print("Temperatura: ");
    //Serial.print(t);
    //Serial.print(" *C ");
    //Serial.print("\n");
  }


  }

  //para controlar el sensor de humedad y temperatura
  if(equipo=="1011")
  {
      //Se hace la lectura analogica del pin A0 (sensor) y se pasa por la
funcion
      //map() para ajustar los valores leidos a los porcentajes que queremos
utilizar
      //lectura 0-->100(%)humedad
      //        1-->0(%)
```

```
      int valorHumedad = map(analogRead(sensorhumedad), 0, 1023, 100, 0);
      mySwitch.send(valorHumedad,24);
      //Serial.print("Humedad: ");
      //Serial.print(valorHumedad);
      //Serial.println("%");
  }

  //para ver que pines de los reles hay activos
  if(equipo=="1100")
  {
    estadorele[0]=digitalRead(4);
    estadorele[1]=digitalRead(5);
    estadorele[2]=digitalRead(6);
    estadorele[3]=digitalRead(7);
    estadorele[4]=digitalRead(8);
    envio=pow(2,4)*estadorele[0]+
pow(2,3)*estadorele[1]+pow(2,2)*estadorele[2]+ pow(2,1)*estadorele[3]+
estadorele[4]+1; //Se le sumam 1 porque si no hay ninguno activo no se podria
mandar el valor 0-->se lo resta en el destino
    mySwitch.send(envio,24);
  }

}

int bintodec(String data)
{
    int acumula=0;
    int k=data.length();
    int l=k-1;
    while (k >= 0)
    {
      float powonente = pow(2,(l-k));
      int dat=(int)data[k];
      if (dat==49)
//El numero 1 como string al pasarlo a entero toma el valor 49 (código ascii)
        dat=1;                                                          //y
el numero 0 el 48
      else
        dat=0;

      acumula=dat*powonente+acumula+0.5*dat;
//el 0.5 hay que introducirlo

//para corregir el error de operar con

//flotantes y truncarlos a enteros
      k=k-1;
    }
    return acumula;
}
```

8.2. Códigos de Raspberry PI

8.2.1 login.php

```
<?php
$password = "a58a14c5390aa81d14063ce53134c24a";
session_start();

if($_POST['password'])
{
    if(md5($_POST['password']) == $password)
    {
        $_SESSION['password'] = "alm";
        $_SESSION['name']="Fran";
    }
    else
    {
        echo "<span style='color:red;font-weight:bold;'>La contraseña es
incorrecta</span>";
    }
}
if(!$_SESSION['password'])
{
?>
<link rel="stylesheet" href="estilo.css">

<h2 align="center">Introduzca su contraseña</h2>

<form class="login" name="form1" method="post" action="<?php echo
$_SERVER['PHP_SELF']; ?>">
<p align="center">
<input class="contraseña" type="password" name="password">
<input class="login" type="submit" name="Submit" value="Login"></p></form>
<?php
}
else
{
?>
<a href="<?php echo $_SERVER['PHP_SELF']; ?>?desconectar=si">Desconectar</a>
<?php
    header("Location: inicio.php");
}
?>
```

8.2.2 inicio.php

```
<?php
session_start();
if($_SESSION['name']!="Fran" )
{
    header("Location: login.php");
    session_destroy();
    exit;

}
?>
<html>
    <head>
        <title>DISPOSITIVOS</title>
        <!enlazamos>
        <link rel="stylesheet" href="estilo.css">
    </head>
```

```
    <body>
        <form action="" method="post"> <!formulario>
        <p><input class= "desconectar" name="Desconectar" value
="Desconectar" type="submit" href="<?php echo $_SERVER['PHP_SELF'];
?>?desconectar=si"></p>
            <h2 align="center">DISPOSITIVOS</h2>
            <p align="center">
                <select name="gadget">
                    <option value="1">Relé  </option>
                    <option value="2">Servo </option>
                    <option value="3">Hum y Tem </option>
                    <option value="4">Humedad del suelo </option>
                    <option value="5">Compruebareles</option>
                    <option value="6">Control raspberry </option>
                    <option
                </select>

            </p>
            <p align="center">
             <input class="enviar" type="submit"   name="Enviar"
value="Enviar"></p>
        <p align="right">
    </body>
</html>

<?php
  if($_POST['Desconectar'] && !$_POST['Enviar'])
  {
    session_destroy();
    header("Location: login.php");
  }
  if ($_POST[gadget]==1 && !$_POST['Desconectar'] && $_POST['Enviar'])
  {
    //redireccionamos a la página que controla los rele
    header("Location: rele.php");
  }

  if ($_POST[gadget]==2 && !$_POST['Desconectar'] && $_POST['Enviar'])
  {
    //redireccionamos a la página que controla los
    //microservos
    header("Location: servo.php");
  }

  if ($_POST[gadget]==3 && !$_POST['Desconectar'] && $_POST['Enviar'])
  {
    //redireccionamos a la página que controla el
    //sensor de temperatura y humedad del aire
    header("Location: humtem.php");
  }

  if ($_POST[gadget]==4 && !$_POST['Desconectar'] && $_POST['Enviar'])
  {
    //redireccionamos a la página que controla el
    //sensor de humedad del suelo
    header("Location: humsuelo.php");
  }
  if ($_POST[gadget]==5 && !$_POST['Desconectar'] && $_POST['Enviar'])
  {
    header("Location: controlareles.php");
  }
  if ($_POST[gadget]==6 && !$_POST['Desconectar'] && $_POST['Enviar'])
```

```
  {
    header("Location: controlrasp.php");
  }
?>
```

8.2.3 rele.php

```
<?php
session_start();
if($_SESSION['name']!="Fran")
{
  session_destroy();
  header("Location: login.php");
  exit;
}
?>
<html>
<head>
  <title>Control de rele</title>
  <!enlazamos>
  <link rel="stylesheet" href="estilo.css">
</head>

<body>
        <form action="" method="post"> <!formulario>
        <p><input class= "desconectar" name="Desconectar" value
="Desconectar" type="submit" href="<?php echo $_SERVER['PHP_SELF'];
?>?desconectar=si">
        <input class="desconectar" type="submit" name="inicio"
value="inicio"></p>
        <h2 align="center"> CONTROL DE RELE</h2>
        <p align="center">
                <select name="relesel">
                    <option value="1">Relé 1</option>
                    <option value="2">Relé 2</option>
                    <option value="3">Relé 3</option>
                    <option value="4">Relé 4</option>
                    <option value="5">Relé 5</option>
                </select>
        </p>
        <p align="center" class="letra">Tiempo encendido(min): <input
class="minutos" type="number"     name="minutos" min="1" max="254"></p>
        <p class="botones" align="center">
        <!botones>
         <input class="encender" type="submit" name="encender"
value="encender">

     <input class="enviar" type="submit" name="apagar"
value="apagar"></p>
</body>
</html>

<?php
if($_POST['Desconectar'] && !$_POST['encender'] && !$_POST['apagar'])
  {
    session_destroy();
    header("Location: login.php");
    exit;
  }
```

```
if ($_POST[encender] && !$_POST['Desconectar'])
{
    $a- exec("sudo python /var/www/html/TransmitRF.py  'eleccion' 'rele'
'$_POST[releselsel]' 'encender' '$_POST[minutos]'");
    echo $a;
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");
}

if ($_POST[apagar] && !$_POST['Desconectar'])
{
    $a- exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'rele'
'$_POST[releselsel]' 'apagar'");
    echo $a;
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");

}
if ($_POST[inicio])
{
    header("Location: inicio.php");
}

?>
```

8.2.4 servo.php

```
<?php
session_start();
if($_SESSION['name']!="Fran")
{
  session_destroy();
  header("Location: login.php");
  exit;
}
?>
<html>
<head>
  <title>Control de microservo</title>
  <!enlazamos>
  <link rel="stylesheet" href="estilo.css">
</head>

<body>
        <form action="" method="post"> <!formulario>
        <p><input class= "desconectar" name="Desconectar" value
="Desconectar" type="submit" href="<?php echo $_SERVER['PHP_SELF'];
?>?desconectar=si">
        <input class="desconectar" type="submit" name="inicio"
value="inicio"></p>
        <h2 align="center"> CONTROL DE MICROSERVO</h2>
        <p align="center">
                <select name="servosel">
                    <option value="1">Microservo 1</option>
                    <option value="2">Microservo 2</option>
                    <option value="3">Microservo 3</option>
                    <option value="4">Microservo 4</option>
                    <option value="5">Microservo 5</option>
                </select>
```

```
        </p>

        <!Vamos a introducir un cuadro para establecer los          <!
gradosde giro del microservo>

        <p class="letra" align="center">Introduzca el ángulo: <input
class="minutos" type="number"     name="angulo" min="0" max="180">
        </p>

        <!botones>
        <p align="center">
         <input class="enviar" type="submit" name="mover" value="mover">


</body>
</html>

<?php
if($_POST['Desconectar'] && !$_POST[mover])
{
    session_destroy();
    header("Location: login.php");
}
if ($_POST[mover] && !$_POST['Desconectar'])
{
    $a- exec("sudo python /var/www/html/TransmitRF.py  'eleccion' 'servo'
'$_POST[servosel]' '$_POST[angulo]'");
    echo $a;
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");
}
if ($_POST[inicio])
{
    header("Location: inicio.php");
}

?>
```

8.2.5 humtem.php

```
<?php
session_start();
if($_SESSION['name']!="Fran")
{
  session_destroy();
  header("Location: login.php");
  exit;
}
?>
<html>
<head>
  <title>Control de temperatura y humedad</title>
  <!enlazamos>
  <link rel="stylesheet" href="estilo.css">
</head>

<body>
        <form action="" method="post"> <!formulario>
```

```
        <p><input class= "desconectar" name="Desconectar" value
="Desconectar" type="submit" href="<?php echo $_SERVER['PHP_SELF'];
?>?desconectar=si">
        <input class="desconectar" type="submit" name="inicio"
value="inicio"></p>
        <h2 align="center"> TEMPERATURA Y HUMEDAD</h2>

        <!botones>
        <p align="center">
         <input class="medirhum" type="submit" name="medirtemp"
value="medirtemp">
         <input class="medirhum" type="submit" name="medirhum"
value="medirhum">
        </p>

</body>
</html>

<?php
session_start();
if($_POST['Desconectar'] && !$_POST[medirtemp] && !$_POST[medirhum])
{
     session_destroy();
     header("Location: login.php");
     exit;
}
if ($_POST[medirtemp] && !$_POST['Desconectar'])
{

    $a=exec("sudo python /var/www/html/TransmitRF.py  'eleccion' 'humtem'
'temperatura'");
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");

    $tem = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    echo "Temperatura ºC:<input type='text' value='$tem'/>";



}
if ($_POST[medirhum] && !$_POST['Desconectar'])
{
    $b=exec("sudo python /var/www/html/TransmitRF.py  'eleccion' 'humtem'
'humedad'");
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    $a- exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");
    $hum = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    echo "Humedad del aire %:<input type='text' value='$hum'/>";
}
if ($_POST[inicio])
{
    header("Location: inicio.php");
}
?>
```

8.2.6 humsuelo.php

```
<?php
session_start();
if($_SESSION['name']!="Fran")
{
  session_destroy();
  header("Location: login.php");
  exit;
}
?>
<html>
<head>
  <title>Medida humedad del suelo</title>
  <!enlazamos>
  <link rel="stylesheet" href="estilo.css">
</head>

<body>
        <form action="" method="post"> <!formulario>
        <p>
        <input class= "desconectar" name="Desconectar" value ="Desconectar"
type="submit" href="<?php echo $_SERVER['PHP_SELF']; ?>?desconectar=si">
        <input class="desconectar" type="submit" name="inicio"
value="inicio"></p>
        <h2 align="center">HUMEDAD DEL SUELO</h2>

        <!botones>
        <p align="center">
         <input class="medirhum" type="submit" name="medirHUM"
value="medirHUM">
        </p>

</body>
</html>

<?php
if($_POST['Desconectar'] && !$_POST[medirHUM])
{
    session_destroy();
    header("Location: login.php");
}
if ($_POST[medirHUM] && !$_POST['Desconectar'])
{

    $a=exec("sudo python /var/www/html/TransmitRF.py  'eleccion'
'humsuelo'");
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");
    $hums = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    echo "Humedad del suelo %:<input type='text' value='$hums'/>";

}
if ($_POST[inicio])
{
    header("Location: inicio.php");
}
?>
```

8.2.7 controlareles.php

```
<?php
session_start();
if($_SESSION['name']!="Fran")
{
  session_destroy();
  header("Location: login.php");
  exit;
}
?>
<html>
<head>
  <title>Estado de los reles</title>
  <!enlazamos>
  <link rel="stylesheet" href="estilo.css">
</head>

<body>
        <form action="" method="post"> <!formulario>
        <p><input class= "desconectar" name="Desconectar" value
="Desconectar" type="submit" href="<?php echo $_SERVER['PHP_SELF'];
?>?desconectar=si">
        <input class="desconectar" type="submit" name="inicio"
value="inicio"></p>
        <h2 align="center">ESTADO DE LOS RELES</h2>

        <!botones>
        <p align="center">
         <input class="comprobar" type="submit" name="Comprobar"
value="Comprobar">
        </p>
<p class="tabla">
        <table id="tabla" align="center">


  <tr>
    <td class="arriba" align="center">Rele 1</td>
    <td class="arriba" align="center">Rele 2</td>
    <td class="arriba" align="center">Rele 3</td>
    <td class="arriba" align="center">Rele 4</td>
    <td class="arriba" align="center">Rele 5</td>
  </tr>
  <tr>
    <td class="abajo" id="1" align="center"></td>
    <td class="abajo" id="2" align="center"></td>
    <td class="abajo" id="3" align="center"></td>
    <td class="abajo" id="4" align="center"></td>
    <td class="abajo" id="5" align="center"></td>
  </tr>

</table>
</p>
</body>
</html>

<?php
if($_POST['Desconectar'] && !$_POST[Comprobar])
{
    session_destroy();
    header("Location: login.php");
```

```
}
if ($_POST[Comprobar] && !$_POST['Desconectar'])
{

    $a=exec("sudo python /var/www/html/TransmitRF.py  'eleccion'
'compruebareles'");
    $reci = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");
    exec("sudo python /var/www/html/TransmitRF.py 'eleccion' 'ack'");
    $mreles = exec("sudo /var/www/html/433Utils/RPi_utils/./RFSniffer");

    $mreles=$mreles-1;

    if ($mreles<0):
    ?>
        <p align="center" class="error">Mensaje de error:Módulos 433MHz con
alimentación insuficiente o no está transmitiendo correctamente</p>
    <?php
    else:
    $binre=decbin($mreles);
    $cadena=(string)$binre;
    while (strlen($cadena) <= 4):
        $cadena= '0' . $cadena;
    endwhile;
?>
    <script type="text/javascript">
    var cadenas = "<?php echo $cadena;?>";

    document.getElementById("1").innerHTML =cadenas[0];
    document.getElementById("2").innerHTML =cadenas[1];
    document.getElementById("3").innerHTML =cadenas[2];
    document.getElementById("4").innerHTML =cadenas[3];
    document.getElementById("5").innerHTML =cadenas[4];
    var x=document.getElementsByClassName("abajo");
    var k=0;
    var dato;
    var str;                        //almacena dato como string
    var n=0;
    while (k<=4)
    {

        dato=x[k].innerHTML;     //para ver el contenido de la celda k de la
tabla
        //document.write(dato);
        n=k+1;
        str=n.toString();
        if (dato==1)
        {
            document.getElementById(str).innerHTML ="On";
        }
        if (dato==0)
        {
            document.getElementById(str).innerHTML ="Off";
        }
        k=k+1;

    }

    </script>
<?php
     endif;
}
if ($_POST[inicio])
```

```
{
    header("Location: inicio.php");
}

?>
```

8.2.8 controlrasp.php

```
<?php
session_start();
if($_SESSION['name']!="Fran")
{
  session_destroy();
  header("Location: login.php");
  exit;
}
?>
<html>
<head>
  <title>Control de Raspberry Pi</title>
  <!enlazamos>
  <link rel="stylesheet" href="estilo.css">
</head>

<body>
        <form action="" method="post"> <!formulario>
        <p><input class= "desconectar" name="Desconectar" value
="Desconectar" type="submit" href="<?php echo $_SERVER['PHP_SELF'];
?>?desconectar=si">
        <input class="desconectar" type="submit" name="inicio"
value="inicio"></p>
        <h2 align="center"> CONTROL DE RASPBERRY PI</h2>

        <!botones>
        <p align="center">
         <input class="reinicioserv" type="submit"
name="reinicioservidor" value="reinicioserv">
        </p>
        <p align="center">
         <input class="reiniciosis" type="submit" name="reiniciosistema"
value="reiniciosis">
        </p>

</body>
</html>

<?php
session_start();
if($_POST['Desconectar'] && !$_POST['reiniciosistema'] &&
!$_POST['reinicioservidor'])
{
     session_destroy();
     header("Location: login.php");
     exit;
}
if ($_POST['reiniciosistema'] && !$_POST['reinicioservidor'])
{

    exec("sudo reboot");
```

```
}
if ($_POST['reinicioservidor'] && !$_POST['reiniciosistema'])
{
    exec("sudo service apache2 restart");
}
if ($_POST[inicio])
{
    header("Location: inicio.php");
}
?>
```

8.2.9 estilo.css

```
body
{
    font-family:"Times New Roman";
    color: white;
    background-color: black;
}

h1
{
    font-size: 90;
}

h2
{
    font-size: 85;
    padding-botton: 30px;
}


.boton2
{
    text-decoration: none;
    padding: 50px;
    width: 100px; /*ancho del boton*/
    font-weight: 600;
    font-size: 40px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
    border: 2px solid #0016b0;
}


select
{
        font-size: 40px;
        border-radius: 100px;
        height: 60px;
        padding: 5px;
        width: 500px;
        margin-bottom: 70px;
}

input.contraseña
{
```

```
  border: 40px;
  border-radius: 40px;
  width: 300px;
  font-size: 30px;
}

input.login
{
    text-decoration: none;
    width: 100px; /*ancho del boton*/
    font-weight: 600;
    font-size: 30px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
    border: 2px solid #0016b0;
}

input.minutos
{
    text-decoration: none;
    width: 100px; /*ancho del boton*/
    font-weight: 600;
    font-size: 30px;
    border-radius: 90px;
    border: 2px solid #0016b0;
}

input.encender
{
    text-decoration: none;
    width: 150px; /*ancho del boton*/
    font-weight: 600;
    font-size: 30px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
    border: 2px solid #0016b0;
}

input.comprobar
{
    text-decoration: none;
    width: 180px; /*ancho del boton*/
    font-weight: 600;
    font-size: 30px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
    border: 2px solid #0016b0;
}

input.medirhum
{
    text-decoration: none;
    width: 200px; /*ancho del boton*/
    font-weight: 600;
    font-size: 30px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
```

```
    border: 2px solid #0016b0;
}

input.enviar
{
    text-decoration: none;
    width: 120px; /*ancho del boton*/
    font-weight: 600;
    font-size: 30px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
    border: 2px solid #0016b0;
}
input.desconectar
{
    text-decoration: none;
    width: 110px; /*ancho del boton*/
    font-weight: 600;
    font-size: 15px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
    border: 2px solid #0016b0;
    text-position: absolute;
}

input.reinicioserv
{
    text-decoration: none;
    width: 250px; /*ancho del boton*/
    font-weight: 600;
    font-size: 40px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
    border: 2px solid #0016b0;
}

input.reiniciosis
{
    text-decoration: none;
    width: 250px; /*ancho del boton*/
    font-weight: 600;
    font-size: 40px;
    color: #ffffff;
    background-color: #1883ba;
    border-radius: 90px;
    border: 2px solid #0016b0;
}

p.botones
{
padding-top:40px;
}

p.letra
{
    font-size:25px;
}
```

```
form.login
{
    padding: 60px;
}

p.tabla
{
    padding:60px;
}

p.error
{
    font-size: 30px;
    color: red;
}

td.arriba, th.arriba {
   background: #1883ba;
   color: #fff;
   width: 90px;
   height:  40px;
   font-size: 20px;
}
td.abajo, th.abajo {
   background: #fff;
   color: #000;
   width: 90px;
   height:  40px;
   font-size: 20px;
}
```

8.2.10 TransmitRF.py

```
import time
import sys
import RPi.GPIO as GPIO
import math

#Funcion para transformar un entero a una cadena de
#caracteres
def binario( decimal ):
    binario=""
    while ( decimal> 0):
        if(decimal %2 ==0):
            binario= "0"+ binario
        else:
            binario= "1" + binario
        decimal=int(math.floor(decimal/2))
    return binario

#La primera parte de eleccion,(que es la cadena a tx)
#es siempre igual.Hay que tx un 0 para que salga del
#reposo, que es tx 1's
eleccion = '111111111110'

#Los 4 siguientes bits identifican al dispositivo a #controlar

#Seccion para controlar el rele
```

```
if sys.argv[2] == "rele":
    releint = int(sys.argv[3])-1 #String->Entero

    #La funcion binario convierte un entero a cadena de
    #caracteres
    bin = binario(releint)

    #le tenemos que llenar hasta 4 para que siempre se
    #tx la misma cantidad de bits. Esto luego nos facilitara
    #la vida a la hora de deshacer el cambio a entero en rx

    while len(bin) < 4:
        bin = '0'+bin

    eleccion= eleccion + bin    #Concatenamos
    #Hay que tx un 1 al final minimo para que transmisor
    #vuelva a reposo
    #si no se le pasan los minutos en el formulario
    #se entiende que esta encendido permanentemente
    if sys.argv[4] == 'encender' and sys.argv[5]=='':
        eleccion = eleccion + '111111111'
        print (eleccion)
    if sys.argv[4] == 'apagar':
        eleccion = eleccion + '000000001'
        print(eleccion)
    if sys.argv[4] == 'encender' and sys.argv[5]!='':
        #Hacemos una regla de 3
        #Tenemos 8 bits para codificar minutos
        minu = int(sys.argv[5])
        minustr= str(binario(minu))
        while len(minustr) < 8:
            minustr = '0' + minustr;
        #Hay que tx un 1 al final minimo para que transmisor
        #vuelva a reposo
        eleccion = eleccion + minustr + '1'

    print(eleccion)
#Seccion para controlar el microservo
if sys.argv[2] == "servo":
    servoint = int(sys.argv[3])+4    #String->int
                                     #5 prim son reles
    servo= binario(servoint)          #int->bin

    while len(servo) < 4:
        servo= '0' + servo

    eleccion= eleccion + servo    #Concatenamos

    #Hacemos una regla de 3
    #Tenemos 8 bits para codificar el angulo
    angulo = int(sys.argv[4])*255
    angulo= angulo /180
    angulostr= str(binario(angulo))

    #Tenemos 8 bits para modular el angulo
    while len(angulostr) < 8:
        angulostr = '0' + angulostr;
    eleccion = eleccion + angulostr
    print (eleccion)
    #Hay que tx un 1 al final minimo para que transmisor
    #vuelva a reposo
    eleccion = eleccion + '1'
```

```
#Seccion para controlar el sensor de humedad y temp.
if sys.argv[2] == "humtem":
    #le mandamos al rx una senal que indica unicamente que
    #tiene que tomar medidas del sensor de temp y hum.
    #En nuestra modulacion este sensor correspondera con el
    #equipo 10
    if sys.argv[3] == "humedad":
        eleccion= '111111111110'+'1010' + '101010101'
    if sys.argv[3] == "temperatura":
        eleccion= '111111111110'+'1010' + '010101011'

if sys.argv[2] == "humsuelo":
    #le mandamos al rx una senal que indica unicamente que
    #tiene que tomar medidas del sensor de hum. suelo
    #En nuestra modulacion este sensor correspondera con el
    #equipo 11
    eleccion= '111111111110'+'1011' + '101010101'

if sys.argv[2] =="compruebareles":
    #le mandamos al rx una senal que indica unicamente que
    #tiene que ver que reles estan activos
    eleccion= '111111111110'+'1100' + '101010101'

if sys.argv[2] =="ack":
    #le mandamos al rx una senal que indica unicamente que
    #tiene que ver que reles estan activos
    eleccion= '111111111110'+'1101' + '010110101'
    print(eleccion);


short_delay = 0.00045
long_delay = 0.00090
extended_delay = 0.0096

NUM_ATTEMPTS = 10
TRANSMIT_PIN = 22

def transmit_code(code):
    '''Transmit a chosen code string using the GPIO transmitter'''
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(TRANSMIT_PIN, GPIO.OUT)
    for t in range(NUM_ATTEMPTS):
        for i in code:
            if i == '1':
                GPIO.output(TRANSMIT_PIN, 1)
                time.sleep(short_delay)
                GPIO.output(TRANSMIT_PIN, 0)
                time.sleep(long_delay)
            elif i == '0':
                GPIO.output(TRANSMIT_PIN, 1)
                time.sleep(long_delay)
                GPIO.output(TRANSMIT_PIN, 0)
                time.sleep(short_delay)
            else:
                continue
        GPIO.output(TRANSMIT_PIN, 0)
        time.sleep(extended_delay)
    GPIO.cleanup()

if __name__ == '__main__':
    for argument in sys.argv[1:]:
```

```
        exec('transmit_code(' + str(argument) + ')')
```

8.2.11 RFSniffer.cpp

```
/*
  RFSniffer

  Usage: ./RFSniffer [<pulseLength>]
  [] = optional

  Hacked from http://code.google.com/p/rc-switch/
  by @justy to provide a handy RF code sniffer
*/

#include "../rc-switch/RCSwitch.h"
#include <stdlib.h>
#include <time.h>       /* clock_t, clock, CLOCKS_PER_SEC */
#include <stdio.h>
#include <unistd.h>
#include <sstream>
#include <string> //should be already included in sstream


RCSwitch mySwitch;



int main(int argc, char *argv[]) {

     // This pin is not the first pin on the RPi GPIO header!
     // Consult https://projects.drogon.net/raspberry-pi/wiringpi/pins/
     // for more information.
     int PIN = 2;
     int bandera=0; //bandera que usaremos para
                    //salir del programa después del primer
                    //dato recibido
     int salvar;    //salvar el valor de la variable recibida
     double t;
     t = (double) clock();            /* inicia el reloj */
     t = t / CLOCKS_PER_SEC;          /*    mide en s    */

     if(wiringPiSetup() == -1) {
       printf("wiringPiSetup failed, exiting...");
       return 0;
     }

     int pulseLength = 0;
     if (argv[1] != NULL) pulseLength = atoi(argv[1]);

     mySwitch = RCSwitch();
     if (pulseLength != 0) mySwitch.setPulseLength(pulseLength);
     mySwitch.enableReceive(PIN);  // Receiver on interrupt 0 => that is pin
#2


     while(bandera==0) {
      t = (double) clock();            /* tiempo que ha pasado */
```

```
      t = t / CLOCKS_PER_SEC;
      if(t>2)                     //si no llega nada en 2s
//salimos del programa
      {
        return 0;
      }
      if (mySwitch.available()) {
        bandera=1; //bandera que nos indica que hemos entrado al bucle y que
no queremos entrar más
        int value = mySwitch.getReceivedValue();

        if (value == 0) {
          printf("Unknown encoding\n");
        } else {

          printf("%i\n", mySwitch.getReceivedValue() );
          salvar=value;
          mySwitch.resetAvailable();
        }

        fflush(stdout);
      }
      usleep(100);

  }

  return salvar;


}
```

9 REFERENCIAS

[1]. (s.f.). Obtenido de https://www.instructables.com/id/Super-Simple-Raspberry-Pi-433MHz-Home-Automation/

10 BIBLIOGRAFÍA

https://hackernoon.com/raspberry-pi-headless-install-462ccabd75d0

https://www.luisllamas.es/configurar-raspberry-pi/

https://geekytheory.com/tutorial-raspberry-pi-crear-servidor-web

http://www.mnp.cl/post/instalar-mysql-php-apache-phpmyadmin-raspberry-pi

https://www.youtube.com/watch?v=4gSF8h4m818

https://www.youtube.com/watch?v=crABHKCK1BQ

https://www.youtube.com/watch?v=PnpEWRdOWDE

https://pimylifeup.com/raspberry-pi-mysql-phpmyadmin/

https://www.raspberrypi.org/forums/viewtopic.php?t=195198

https://www.prometec.net/raspberry-servidor-internet/

https://www.youtube.com/watch?v=3CtcquGMzvY

http://www.telematica.utfsm.cl/telematica/site/artic/20141001/asocfile/20141001143043/tutorial_proyecto_raspberry_pi.pdf

https://victorhckinthefreeworld.com/2018/07/25/configurar-un-certificado-ssl-autofirmado-para-mi-raspbery-pi-con-raspbian/

https://www.digicert.com/es/ayuda/

https://www.w3.org/Style/Examples/011/firstcss.es.html

https://vinkula.com/crear-botones-personalizados-con-html5-y-css3/

http://elpelucasapbe.foroactivo.com/t34-selectores-html-y-php

https://www.princetronics.com/how-to-read-433-mhz-codes-w-raspberry-pi-433-mhz-receiver/

https://github.com/ninjablocks/433Utils/issues/27

http://www.madnesselectronics.com/tutorial-sensor-de-humedad-de-suelo/

https://norfipc.com/codigos/como-proteger-pagina-web-contrasena-php.php

https://omicrono.elespanol.com/2017/07/como-funciona-cifrado-md5/

https://codeforgeek.com/2014/08/session-handling-php/

http://informaticosimpatico-linux.blogspot.com/2014/11/mostrar-errores-en-php-desde-apache.html

https://www.w3schools.com/tags/tag_script.asp

http://www.forosdelweb.com/f13/obtener-datos-tabla-para-procesarlos-por-javascript-877091/

https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.bpxbd00/clock.htm

https://www.instructables.com/id/Arduino-433Mhz-Wireless-Communication-Rc-Switch/

https://jadasupport.wordpress.com/2014/09/29/arduino-controlando-salidas-electricas-via-433mhz/

https://www.raspberrypi.org/forums/viewtopic.php?f=37&t=66946

http://wiringpi.com/download-and-install/

https://www.instructables.com/id/Super-Simple-Raspberry-Pi-433MHz-Home-Automation/

https://www.google.com/search?q=connect+a+relay&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiMsYKS_-fdAhUB-qQKHQl5CaIQ_AUICigB&biw=1366&bih=662#imgrc=sfoH-n6n-cMwZM:

https://components101.com/5v-relay-pinout-working-datasheet

https://programarfacil.com/blog/arduino-blog/sensor-de-nivel-de-agua-con-arduino/

https://playground.arduino.cc/Code/SimpleTimer

https://programarfacil.com/tutoriales/fragmentos/servomotor-con-arduino/

http://www.omniblug.com/sensor-temperatura-humedad-DHT11-DHT22.html