Universidad Miguel Hernández de Elche

MASTER UNIVERSITARIO EN ROBÓTICA

“Automatización de un sistema de recirculación de ACS sin tubería de retorno mediante equipos Raspberry Pi”

Trabajo de Fin de Máster

Curso académico 2020 – 2021

Autor: Carlos Hernández Iglesias

Tutor: Carlos Pérez Vidal

INDICE GENERAL

INDICE DE ILUSTRACIONES

INDICE DE TABLAS

AGRADECIMEINTOS

Me gustaría expresar mi agradecimiento a todos aquellos que, de una u otra forma, han hecho posible la realización de este trabajo.

Quiero agradecer a mi director de trabajo, Carlos Pérez Vidal, su implicación y apoyo para desarrollar el trabajo. Igualmente, a Oscar Reinoso y al departamento de Ingeniería de Sistemas y Automática, por su disposición y ayuda.

También agradecer a mi mujer su apoyo y a mis hijos el sacrificio, sin ellos no habría sido posible terminar esta labor.

RESUMEN

El presente trabajo, trata de llevar a cabo el diseño y programación de un sistema de recirculación de agua caliente, en viviendas en las que no es posible la instalación de una tubería de retorno, mediante microcontroladores Raspberry Pi Zero.

Primeramente, se revisa el estado del arte en cuanto a los sistemas existentes en el mercado, y a las posibles opciones de microcontroladores que existen con los que es posible realizar este sistema.

Tras esta primera investigación, se estudiaron las diferentes opciones de montaje del sistema, contemplando las ventajas e inconvenientes de cada uno, bajo el punto de vista funcional y económico, principalmente.

Una vez escogido el esquema de montaje de los elementos, fue necesario el aprendizaje sobre la puesta en marcha y programación de estos microcontroladores en Python 3. Este aprendizaje se llevó a cabo con tutoriales de diferentes páginas web, incluyendo los existentes en la página oficial de Raspberry Pi además de otras páginas web de la multitud que hay publicadas.

Una vez adquiridos estos conocimientos, se realizó la programación de tres sistemas de conexionado diferentes, uno cableando la vivienda desde un punto común (en el que iría el controlador), otro incorporando tres microcontroladores (uno por punto de consumo y otro en la generación de A.C.S.) y por último se añadieron a este esquema sensores de temperatura en los puntos de consumo para el control de la recirculación.

La programación del sistema mediante tres microcontroladoras diferidas requirió la conexión de estas a una Red Wifi doméstica, y la comunicación entre ellas mediante peticiones y consultas Web.

Una vez realizado el conexionado y programación de todos los elementos, se procedió a probar el sistema, de forma únicamente eléctrica, es decir, sin conectar los elementos terminales (electroválvulas, bomba, etc.)

Palabras clave: Raspberry Pi, Recirculación de ACS, Recirculación de agua caliente sanitaria, Ahorro de agua potable, Python.

ABSTRACT

The present work tries to carry out the design and programming of a hot water recirculation system, in residences where it is not possible to install a return pipe, using Raspberry Pi Zero microcontrollers.

First of all, the state of the art has been reviewed attending to the existing systems on sale, and the existing microcontroller options to afford the goal.

After this first investigation, different scheme options of the system were studied, considering pros and cons of each one, paying special attention to the functional and economic aspect.

Once the assembly scheme of the elements had been chosen, it was necessary to learn about the start-up and programming of these microcontrollers in Python 3. This learning was carried out with tutorials on different web pages, including those on the official website of Raspberry Pi.

Once this knowledge was acquired, the programming of three different schemes of microcontroller connection were carried out, one wiring the residence from a common point, another one incorporating three microcontrollers (one per consumption point and another in the hot water generator) and finally temperature sensors were added at consumption points to control recirculation.

Programming the system through three deferred microcontrollers required their connection to a home Wi-Fi network, and communication between them through requests and Web queries.

Once the connection and programming of all the elements had been carried out, the system was tested, only electrically, that means, without connecting terminal elements (solenoid valves, pump, etc.)

Keywords: Raspberry Pi, DHW recirculation, Hot water recirculation, saving drinking water, Python.

1. INTRODUCCIÓN

1.1. Motivación

La motivación del presente trabajo es tanto de carácter medioambiental, como económico.

La escasez de agua en gran parte de la península ibérica, y especialmente en la zona sureste, es un problema bien conocido y con muchas vías de mejora, tanto institucional como en el ámbito industrial y del hogar.

En lo que se refiere al hogar, la mayoría de las mejoras en el consumo de agua potable, van encaminadas a la reducción de su consumo mediante diversas vías, como pueden ser:

- Reducción de caudales en grifos mediante aireadores
- Electrodomésticos más eficientes
- Concienciación social:

Una de las opciones de reducción de consumo, en la que no se hace hincapié por parte de la normativa vigente, es la recirculación de ACS (Agua Caliente Sanitaria) en las viviendas de nueva construcción.

La regulación actual, se refiere a este sistema en el Código Técnico de la Edificación (CTE) en su documento básico DB-HS (Salubridad), en el que se indica lo siguiente (punto 3.2.2.1):

“Tanto en instalaciones individuales como en instalaciones de producción centralizada, la red de distribución debe estar dotada de una red de retorno cuando la longitud de la tubería de ida al punto de consumo más alejado sea igual o mayor que 15 m.”

Dado que las viviendas de nueva construcción difícilmente superan los 100m^2 de superficie útil, raramente superan los 15 metros entre el punto de generación y el punto de consumo más alejado.

Si a todo esto, añadimos que, en caso de las calderas instantáneas, ya sean de gas, u otro combustible, tenemos un tiempo de retraso entre la activación del interruptor de flujo y la estabilización de la temperatura del agua a la salida de esta, tenemos una cantidad de agua que se desperdicia, hasta que obtenemos la temperatura deseada.

La solución más extendida a este problema reside en la implementación de un sistema de retorno con una tubería adicional, que mantiene agua caliente a la temperatura de consigna, en la entrada de todos los cuartos húmedos de la vivienda. Este sistema, tiene como ventaja el ahorro de energía, pero, por el contrario, produce un incremento del consumo de energía para el mantenimiento de la temperatura del circuito.

El problema de consumo de agua se solventaría incorporando la recirculación mediante la tubería de retorno mencionada, pero esto no es posible en una vivienda ya construida, debido a que, para ello sería necesario la demolición de los falsos techos de la vivienda para pasar dicha tubería de retorno.

1.2. Objetivos

El presente trabajo, tiene como objetivo principal, el diseño de un sistema de recirculación de ACS, teniendo en cuenta la imposibilidad de incorporar una tubería adicional, tal y como pasa en una gran cantidad de las viviendas construidas en los últimos años.

El sistema constará de una parte mecánica y otra de control.

La parte mecánica estará compuesta de las válvulas, tuberías, equipos de impulsión y demás elementos necesarios para el correcto funcionamiento del sistema.

La parte de control se realizará mediante equipos Raspberry Pi, además de los actuadores y sensores necesarios para el funcionamiento.

El objetivo principal del trabajo reside en el aprendizaje del lenguaje Python, compatible con los equipos Raspberry Pi, y la familiarización con estos equipos dadas las posibilidades de estos para la automatización y su utilización en proyectos de robótica con baja necesidad de capacidad de procesamiento.

El objetivo principal del sistema diseñado será conseguir un ahorro de agua en el uso normal de la instalación de ACS, reduciendo al máximo el agua desperdiciada.

Como objetivo secundario, se contempla la reducción de consumo energético respecto a un sistema convencional de retorno de ACS gracias a la automatización y las posibilidades que brinda un sistema programado mediante una o varias Raspberry Pi.

1.3. Resumen del Trabajo

En el presente trabajo, trata de llevar a cabo el diseño y programación de un sistema de recirculación de agua caliente, en viviendas en las que no es posible la instalación de una tubería de retorno, mediante microcontroladores Raspberry Pi Zero.

Se trata de elaborar un sistema que logre un ahorro en el consumo de agua, reduciendo el desperdicio de la misma cuando se espera que el agua caliente sanitaria llegue al punto de consumo deseado.

La elaboración de este sistema se realiza mediante equipos electromecánicos y microcontroladores de bajo coste, tratando de obtener una alternativa a los equipos comerciales que tienen un coste relativamente elevado.

El principal aprendizaje obtenido en el trabajo es la familiarización con el lenguaje de programación Python, por otro lado, se han adquirido conocimientos sobre algunas de las librerías de los microcontroladores Raspberry Pi y las posibilidades que ofrecen estos equipos para la automatización de sistemas.

2. ESTADO DEL ARTE

2.1. INTRODUCCIÓN

En la mayoría de las viviendas que, por tamaño, no tienen una distancia entre la generación de ACS y el último consumo, superior a 15 m, tal y como indica el Código Técnico de la Edificación (CTE) , disponen no disponen de ningún sistema de recirculación de agua, que evite el desperdicio de la misma. Estos sistemas tienen el esquema que se indica en la figura siguiente:

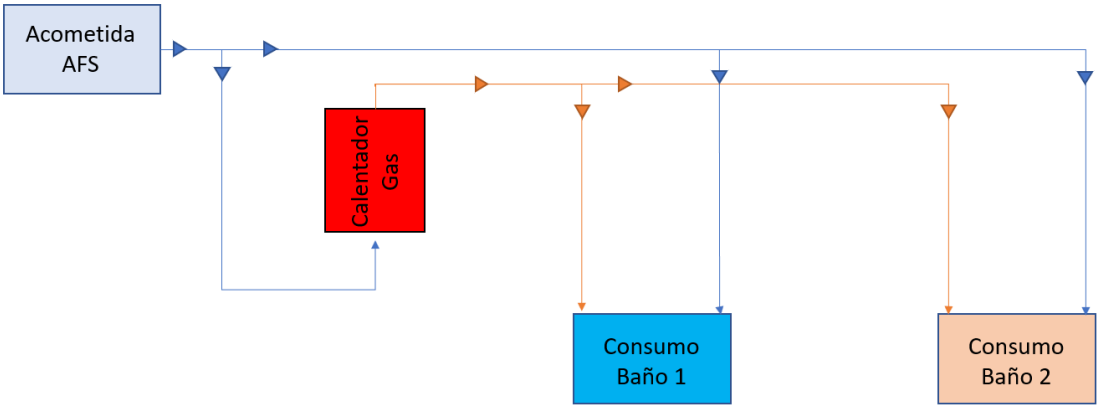

Figura 2.1 Esquema tipo ACS para vivienda sin recirculación

Tradicionalmente, para evitar el desperdicio de agua se han utilizado sistemas de recirculación de agua caliente sanitaria con tubería adicional. Esta recirculación, tal y como se muestra en la siguiente figura.

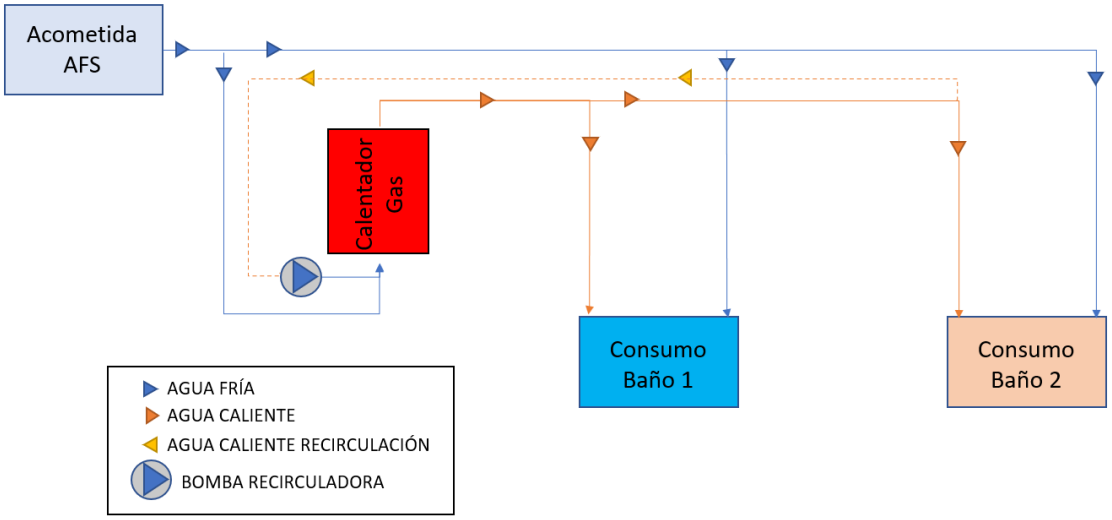

Figura 2.2 Esquema tipo de instalación con tubería de retorno.

Sería necesario realizar más pruebas en diferentes viviendas para poder comparar el desperdicio de agua debido a la no disposición de sistemas de retorno de ACS. Sin embargo, realizando una medición en una vivienda tipo de 100m2, con dos baños a diferentes distancias de la generación de ACS, se han

obtenido los siguientes datos de desperdicio de agua hasta la obtención de esta a la temperatura deseada:

Local Húmedo	Agua Desecho Hasta Tª de Consigna (litros)	Nº usos al día	Litros diarios	m3/ Trimestre	m3/ año
Baño 1	7,5	3	22,5	2,05	8,21
Baño 2	5	3	15	1,37	5,48
Cocina	1,5	3	4,5	0,41	1,64
Lavadero	1	2	2	0,18	0,73
Total		11	44	4,02	16,06

Tabla 2.1 Desperdicio de agua para familia de 4 personas

Como se puede observar, la cocina y el lavadero, por estar situados muy cerca del punto de generación de ACS, no contribuyen en exceso al desperdicio de agua. Tan solo 0,59 m3/trimestre (2,36 m3/año, según el patrón de consumo contemplado).

Para tener en cuenta el coste económico de este desecho, hay que tener muchos factores en cuenta, como el lugar dónde esté situada la vivienda. En este caso, se ha realizado un cálculo con los datos de Elche. Para ello, teniendo en cuenta el consumo medio de una familia de 4 miembros, según se indica en el informe de desarrollo sostenible 2019 (Aigües d'Elx):

Consumo de agua Según Aigües d'Elx	
Consumo diario por persona (L)	110,7
Consumo diario Familia de 4 personas (L)	442,8
Consumo por familia y trimestre (m3)	40,41
Consumo por familia y año (m3)	161,62

Tabla 2.2 Consumo medio Familia de 4 miembros según Informe de desarrollo sostenible (Aigües d'Elx)

Y teniendo en cuenta el coste del agua, y del tramo de consumo que hablamos, podemos ver que la diferencia en el coste anual de ahorrarse el desperdicio de agua caliente en todos los puntos de consumo de un hogar tipo, es de 44,10 €/año:

Año	Lectura Contador	Agua Servic	Agua Consumo	Alquiler Contador	Alcantar. Servicio	Alcantar. Consumo	Basura Servicio	Saneam Servicio	Saneam Consumo	IVA	TOTAL
Trimestre normal	40,41	13,15	41,30	2,28	5,005	5,04	12,48	11,21	17,82	6,93	**115,22**
Trimestre Con Sistema de Recirc.	36,39	13,15	33,65	2,28	5,005	4,22	12,48	11,21	16,05	6,08	**104,12**
									Diferencia Trimestral (€)		**11,10**
									Diferencia Anual (€)		**44,40**

Tabla 2.3 Ahorro consumo de agua en factura. Todos los consumos

Y en caso de que se trate únicamente de los dos puntos más alejados de la vivienda (en este caso los baños) será de 37,77 €/año:

Año	Lectura Contador	Agua Servic	Agua Consumo	Alquiler Contador	Alcantar. Servicio	Alcantar. Consumo	Basura Servicio	Saneam Servicio	Saneam Consumo	IVA	TOTAL
Trimestre normal	40,41	13,15	41,30	2,28	5,005	5,04	12,48	11,21	17,82	6,93	**115,22**
Trimestre Con Sistema de Recirc.	36,99	13,15	34,79	2,28	5,005	4,34	12,48	11,21	16,31	6,21	**105,78**
									Diferencia Trimestral (€)		**9,44**
									Diferencia Anual (€)		**37,77**

Tabla 2.4 Ahorro consumo de agua en factura. Solo consumos más alejados.

Como se puede observar, la diferencia entre incluir o no los puntos más cercanos al punto de generación, es muy bajo (6,63€/año).

2.2. SISTEMAS DE RECIRCULACIÓN AGUA CALIENTE SANITARIA SIN TUBERÍA DE RETORNO

Actualmente, para solucionar el problema de desperdicio de agua en sistemas sin tubería de retorno, existen diversas soluciones comerciales de diferentes tipos que retornan el agua al circuito de agua fría hasta que se dispone de agua caliente en el punto de consumo.

Por un lado, están los sistemas de recirculación que disponen una bomba en cada cuarto húmedo de la vivienda (cocina, aseos, etc.) y recirculan el agua hasta obtener una temperatura de consigna en el cuarto húmedo.

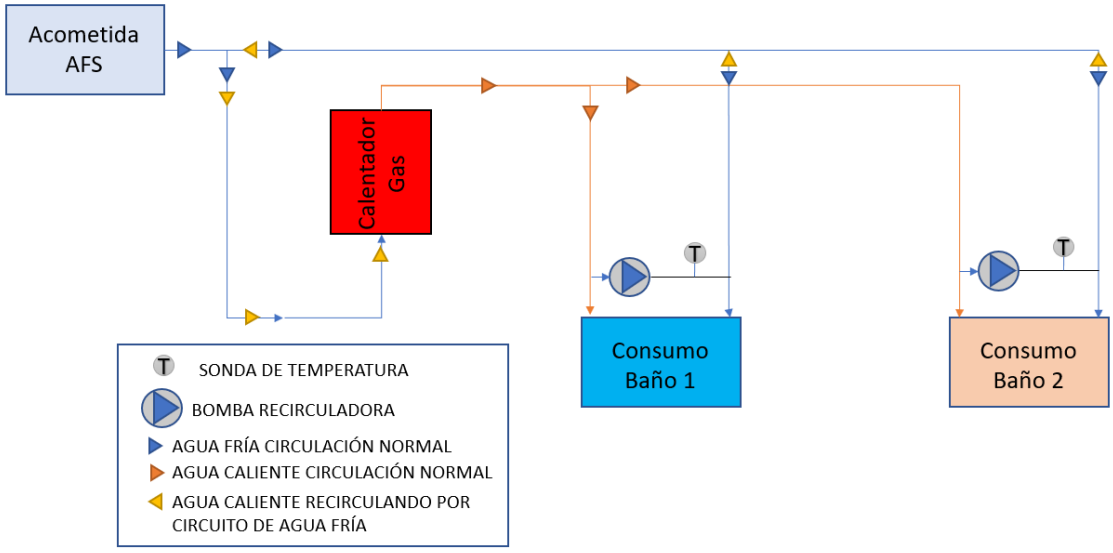

Figura 2.3 Sistemas de recirculación sin tubería de retorno con bomba independiente

Entre los equipos comerciales está Aqua Return. Este sistema, tiene un coste de 347€ /ud (www.aquareturn.com), lo que para una vivienda tipo con dos baños (suponiendo que la cocina está situada cerca de la producción de ACS), supone un coste de 694€.

Figura 2.4 Sistema Aqua Return

Este sistema presenta la ventaja de tener un funcionamiento independiente, sin comunicaciones y de no necesitar ningún elemento adicional en la caldera o calentador.

Por otro lado, están los sistemas que disponen de una bomba común a todos los cuartos húmedos, situada en las proximidades de la generación de ACS y módulos individuales en cada cuarto húmedo donde se necesita realizar la recirculación, con un bypass que permite el paso de agua al circuito de agua fría cuando se necesita.

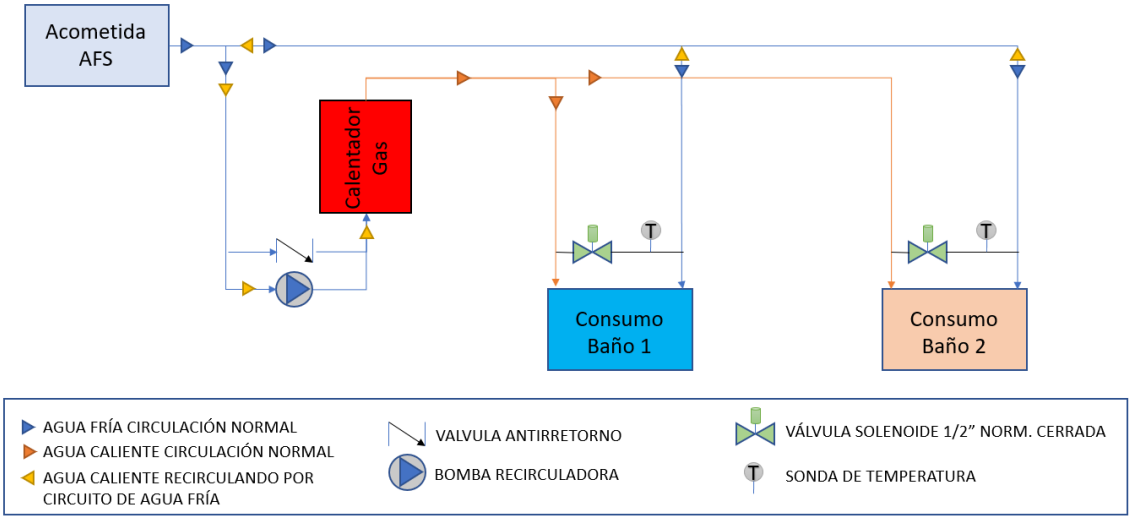

Figura 2.5 Sistemas de recirculación sin tubería de retorno con bomba común

La gran ventaja de este sistema es la utilización de una bomba recirculadora común, lo que, en principio, debería resultar más económico, tratándose del elemento que más encarece el sistema.

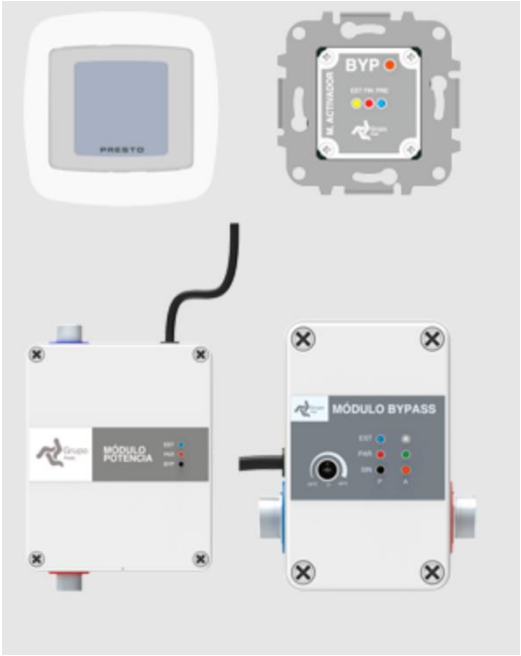

Figura 2.6 Sistema Presto Go System (Prestoiberica)

La gran ventaja de este sistema, debería ser el precio, sin embargo, según se puede ver en la tarifa de la marca (Tarifa Presto) el coste de este sistema para una vivienda con dos puntos de recirculación es de 931,80€ como mínimo.

2.3. MICROCONTROLADORES DE BAJO COSTE

En la actualidad, existen muchos sistemas de bajo coste que se pueden utilizar para automatizar procesos sencillos. Dependiendo de la complejidad, y las necesidades de procesamiento, las alternativas pueden ser diferentes. A continuación, se mencionan los equipos principales existentes en la actualidad que cumplen con estos requisitos.

2.3.1. Raspberry Pi

Raspberry Pi, es un pequeño ordenador, desarrollado en Reino Unido por la fundación Raspberry Pi, cuyo objetivo era principalmente educativo.

El proyecto comenzó en 2006 en la Universidad de Cambridge, y su objetivo principal era el aprendizaje de la programación en niños.

El primer modelo comercial se empezó a vender en el año 2012, y hoy en día, existen varios modelos que, según sus características, se pueden utilizar para diferentes aplicaciones:

2.3.1.1. Raspberry Pi Pico

Se trata de un microcontrolador de poco menos de 5€ de coste, sin posibilidad de comunicaciones (Wifi o Ethernet), programable en C/C++ y Micro Python.

Figura 2.7 Placa Raspberry Pi Pico

Sus características principales son:

- Procesador Dual-Core ARM Cortex M0+ a 133 MHz
- 264kB de SRAM y 2MB de memoria Flash
- Soporte USB 1.1 Host y Device
- Soporte de bajo consumo "Sleep"
- 26 pines GPIO (General Purpose Input Output)
- 3 entradas analógicas (ADC) de 12-bit

Esta placa no dispone de sistema operativo, y no puede más que realizar un único programa que hayamos cargado previamente desde un ordenador.

Mediante los pines GPIO se pueden controlar entradas y salidas digitales, lo que lo hace muy adecuado para sistemas simples que no necesitan de conectividad.

2.3.1.2. Raspberry Pi Zero / W / H

En este caso, sí que estamos hablando de un miniordenador, ya que se trata de un equipo con sistema operativo, capaz de ejecutar varios procesos al mismo tiempo, y más complejos debido a su mayor capacidad de procesamiento respecto a las placas Pico.

El sistema operativo es específico para los equipos Raspberry Pi, y se denomina Raspberry Pi OS (anteriormente se denominaba Raspbian). Se trata de una distribución del sistema operativo GNU/Linux basado en Debian (Software Libre).

También se comercializan Raspberry Pi Zero con Wifi y Bluetooth (W) y con pines pre-soldados en la placa (H), lo cual facilita mucho el prototipado de los proyectos.

El modelo más sencillo tiene un coste aproximado de menos de 10 € hasta los 16€ del modelo con Wifi y conectores soldados.

Las características técnicas del equipo son idénticas en todas sus versiones:

- Procesador integrado de un núcleo ARM 11 Broadcom BCM2835 @ 1 GHz
- RAM: 512 MB
- Lector de tarjetas micro-SD
- Salida mini HDMI
- Dos puertos micro USB (uno de ellos utilizado para alimentación)
- 26 pines GPIO (General Purpose Input Output)

Figura 2.8 Raspberry Pi Zero

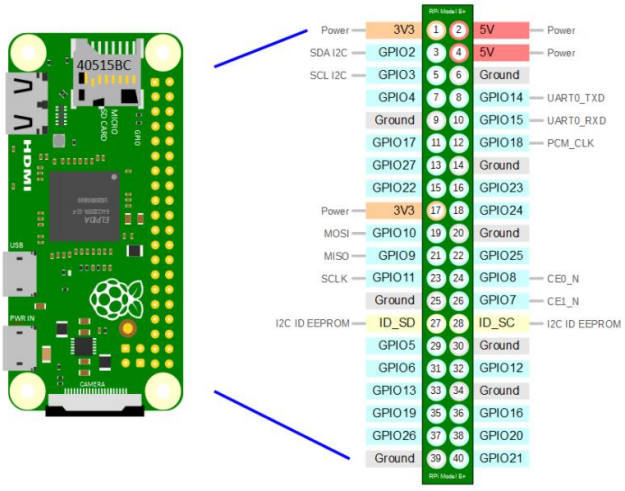

Figura 2.9 Esquema conectores Raspberry Pi Zero

Se trata de una placa multitarea, por lo que es posible ejecutar varios procesos a la vez, con la gran ventaja de la disponibilidad de comunicación, tanto Wifi como Bluetooth.

2.3.1.3. Raspberry Pi 1/2/3/4

Se trata del sistema original de Raspberry Pi, el cual ha ido evolucionando con los años hasta el más actual (Raspberry Pi 4).

Las características han ido evolucionando, pero salvo en los primeros modelos, todos disponen de conexión Ethernet, Wifi y Bluetooth. También disponen de mayor capacidad de procesamiento que las placas Zero, antes comentadas.

En la siguiente tabla se muestran las características de estos equipos según el modelo.

Especificaciones	Raspberry Pi 1 modelo A (descontinuada)	Raspberry Pi 1 modelo B (descontinuada)	Raspberry Pi 1 modelo B+	Raspberry Pi 2 modelo B	Raspberry Pi 3 modelo B	Raspberry Pi 3 modelo B+	Raspberry Pi 3 modelo A+	Raspberry Pi 4 modelo B
SoC (CPU, GPU, DSP, RAM y puertos USB)	Broadcom BCM2835			Broadcom BCM2836	Broadcom BCM2837			Broadcom BCM2711
CPU	ARM 1176JZF-S a 700 MHz (familia ARM11)			900 MHz quad-core ARM Cortex A7	1.2GHz 64-bit quad-core ARMv8	1.4GHz 64-bit quad-core ARMv8		1.5GHz 64-bit quad-core Cortex-A72
Juego de instrucciones	RISC de 32 bits				RISC de 64 bits			
GPU	Broadcom VideoCore IV, OpenGL ES 2.0, MPEG-2 y VC-1 (con licencia), 1080p30 H.264/MPEG-4 AVC							Broadcom VideoCore VI, OpenGL ES 3.0, 1080p30 H.264/MPEG-4 AVC, 4kp60 H.265
Memoria	256 MB (compartidos con la GPU)	512 MiB (compartidos con la GPU)		1 GB (compartidos con la GPU)			512 MiB (compartidos con la GPU)	1 GB, 2 GB, 4 GB u 8 GB (compartidos con la GPU)
Puertos USB 2.0	1	2 (vía hub USB integrado)	4				1	2
Puertos USB 3.0	Ninguno							2
Entradas de vídeo	Conector MIPI CSI que permite instalar un módulo de cámara desarrollado por la Raspberry Pi Foundation							
Salidas de vídeo	Conector RCA (PAL y NTSC), HDMI (rev1.3 y 1.4), Interfaz DSI para panel LCD							Conector RCA (PAL y NTSC), microHDMI rev. 2.0, Interfaz DSI para panel LCD
Salidas de audio	Jack de 3.5 mm, HDMI							Jack de 3.5 mm, 2 puertos microHDMI
Almacenamiento	SD / MMC / ranura para SDIO			MicroSD				
Conectividad de red	Ninguna	10/100 Ethernet (RJ-45) vía hub USB			Puerto RJ-45 (ethernet) de 10/100Mbps vía hub USB Wi-Fi 802.11bgn Bluetooth 4.1	Puerto RJ-45 (Ethernet) de 10/100/1000Mbps vía hub USB limitado a 300Mbit/s Wi-Fi 802.11ac de doble banda Bluetooth 4.2 BLE	Wifi 802.11ac de doble banda Bluetooth 4.2 BLE	Puerto RJ-45 10/100/1000Mbps vía hub USB 3.0 Wi-Fi 802.11ac de doble banda Bluetooth 5.0 BLE
Periféricos de bajo nivel	8 x GPIO, SPI, I²C, UART			17 x GPIO y un bus HAT ID				
Consumo energético	500 mA (2.5 W)	700 mA (3.5 W)	600 mA (3.0 W)	800 mA (4.0 W)				Máximo 3A (15.3 W)
Fuente de alimentación	5 V vía Micro USB o puerto GPIO							5 V vía USB-C o puerto GPIO
Dimensiones	85mm x 53mm							
Sistemas operativos soportados	GNU/Linux: Raspbian, Fedora (Pidora), Arch Linux (Arch Linux ARM), Slackware Linux, SUSE Linux Enterprise Server for ARM, RISC OS							GNU/Linux: Raspbian

Figura 2.10 Características Raspberry Pi (Wikipedia)

2.3.2. Arduino

Arduino es una placa electrónica de hardware libre que utiliza un microcontrolador reprogramable con una serie de pines que permiten establecer conexiones entre el controlador y los diferentes sensores y actuadores.

La plataforma Arduino se programa mediante el uso de un lenguaje propio basado en el lenguaje de programación de alto nivel Processing que es similar a C++.

Arduino está basado en C y soporta todas las funciones del estándar C y algunas de C++.

El proyecto «Arduino» se inició en el año 2005 como un proyecto enfocado a estudiantes en el Instituto IVREA (IDII), en Ivrea (Italia), para abaratar los costes de los microcontroladores que utilizaban por entonces.

Se trata de un hardware libre y la mayoría de las placas Arduino constan de un microcontrolador AVR Atmel-8 bits, cada microcontrolador consta de diversas cantidades de memoria flash, pines y funciones. Las placas utilizan pines/cabezales hembra de una o dos hileras que facilitan las conexiones e incorporación en otros circuitos.

Las placas Arduino pueden conectarse con módulos adicionales denominados shields, que aumentan las características técnicas de la placa debido a que poseen circuitos específicos que añaden una o más funcionalidades extras a la placa Arduino nativa en la cual se utilice, también se les conoce como placas de expansión. La mayoría de estos shields se conectan a través de un bus serie I²C, aunque existen también aquellas que emplean conexión mediante el bus UART (Universal Asynchronous Receiver-Transmitter, por su traducción al español Transmisor-Receptor Asíncrono Universal), así como con el bus SPI (Serial Peripheral Interface, por su traducción al español Interfaz Periférica Serie). (Wikipedia)

Las placas Arduino no incorporan Wifi, aunque es posible su incorporación, pero para ello, es necesario programar dicha comunicación, lo que complica ciertas aplicaciones.

Figura 2.11 Arduino UNO

2.3.3. Otros

La incorporación del IoT o Internet de las cosas, ha hecho que salgan al mercado multitud de microcontroladores de bajo coste, que pueden cubrir tareas sencillas.

Unos de los más utilizados en la actualidad son las placas basadas en el SoC ESP8266, que tiene incorporado Wifi totalmente funcional.

El ESP8266 normalmente viene integrado en un módulo. Esto es debido a que el propio SoC ESP8266 no tiene memoria Flash integrada. El primero que vio la luz fue el ESP-01 el cual estaba pensado para funcionar como interfaz Wifi de las placas de Arduino.

A partir de este módulo surgieron muchos más hasta que finalmente irrumpió en el mercado el ESP-12, el más popular de todos los módulos. Este módulo se utiliza en multitud de placas siendo las más famosas NodeMCU y Wemos.

Estas placas se programan en diferentes lenguajes, Arduino, Lua, MicroPython, C/C++, Scratch, etc.

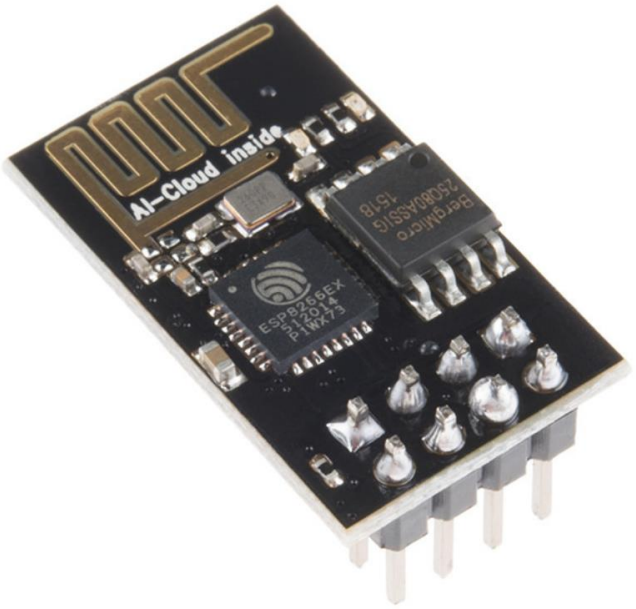

Figura 2.12 Microcontrolador Basado en ESP8266

3. METODOLOGÍA

3.1. ESTUDIO ECONÓMICO DE LAS DIFERENTES SOLUCIONES COMERCIALES

Como se ha visto en apartados anteriores, el ahorro obtenido del precalentamiento del agua es de un máximo de 44,40€ en el caso de realizarlo en todos los puntos de consumo de agua caliente de la casa, y de 37,77€. Puesto que el coste de cualquier sistema comercial se incrementa notablemente en función de los puntos a instalar, vamos a tomar como base para el cálculo de la amortización, el coste de implementar un sistema de recirculación en los 2 baños de la vivienda únicamente.

	Kit Básico para un punto de Consumo	Precio por Punto de Consumo Adicional	Coste Para 2 Puntos de Consumo	Ahorro Anual	Periodo de Amortización en años
Aqua Return	347,00 €	347,00 €	694,00 €	37,77 €	18,37
Presto Go	599,50 €	332,30 €	931,80 €	37,77 €	24,67

Tabla 3.1 Periodo de amortización para los 2 puntos de consumo más alejados

Como se puede ver, los periodos de amortización muy altos, ya que, dado que la garantía de dichos productos es de 2 años, tendremos como mínimo 16 años en los que el producto deba estar funcionando sin averías.

Evidentemente hay también que contemplar el beneficio ecológico del ahorro de agua, que sumado al económico da una perspectiva más favorable a la instalación de estos sistemas.

En caso de instalarse el sistema en 4 puntos de consumo, el coste de los equipos sube en gran medida, pero el ahorro anual tan solo supone 6,63€ anuales. Por este motivo, el periodo de amortización de los equipos adicionales supera los 100 años en ambos casos.

	Kit Básico para un punto de Consumo	Precio por Punto de Consumo Adicional	Coste Adicional Para 4 Puntos de Consumo	Ahorro Anual	Periodo de Amortización en años
Aqua Return	347,00 €	347,00 €	694,00 €	6,63 €	104,73
Presto Go	599,50 €	332,30 €	664,60 €	6,63 €	100,30

Tabla 3.2 Periodo de amortización para los puntos de consumo adicionales cercanos al sistema de producción de ACS.

3.2. SOLUCIÓN ALTERNATIVA A LOS SISTEMAS COMERCIALES

Debido al coste de los sistemas comerciales, se ha contemplado la opción de realizar la función de estos sistemas, mediante equipos electromecánicos convencionales:

- Válvulas
- Solenoides
- Bombas Recirculadoras

Todo ello comandado por equipos que puedan automatizar la operación de los mismos, con un coste significativamente inferior a lo que supone un equipo comercial.

Como elemento de control de los diferentes elementos, se toma la placa Raspberry Pi Zero WH, con Wifi y conectores soldados, por las siguientes razones:

1. Raspberry Pi se programa en Python, que como se ha comentado en el apartado de motivación, es uno de los objetivos de este trabajo
2. Se trata de una placa de prestaciones suficientes para manejar un sistema como este.
3. Dispone de comunicaciones inalámbricas, lo cual permite la interconexión de elementos sin necesidad de cableado.

Se contemplan varias formas de realizar la instalación:

3.2.1. Sistema autónomo en cada baño

Este planteamiento contempla la colocación de una bomba recirculadora y una válvula solenoide en cada baño con una placa controladora que se active a través de un botón.

En este caso concreto, resultaría posible la utilización del modelo Raspberry Pi Pico, siempre y cuando no necesitemos activar el sistema mediante una aplicación móvil, o bien una página web.

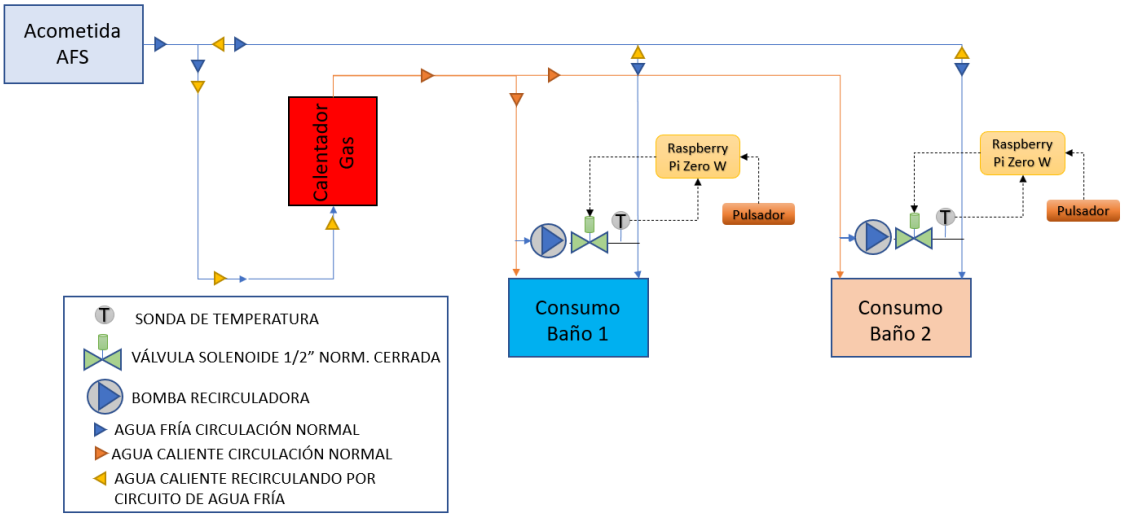

Figura 3.1 Recirculación Independiente por Baño

Como ventaja de este sistema, es la sencillez y que no es necesario ningún tipo de comunicación. Cada sistema es independiente.

El mayor inconveniente, es la necesidad de instalar una bomba recirculadora en cada punto de consumo, lo que supone un coste mayor.

3.2.2. Sistema de control Centralizado

La segunda opción contemplada reside en colocar una única placa controladora centralizada que controle todos los elementos mediante cableado entre las diferentes zonas. En nuestro caso:

- Caldera
- Baño 1
- Baño 2

La zona del lavadero y la cocina, se desestiman debido al criterio coste/beneficio de implantar este sistema en dichas estancias.

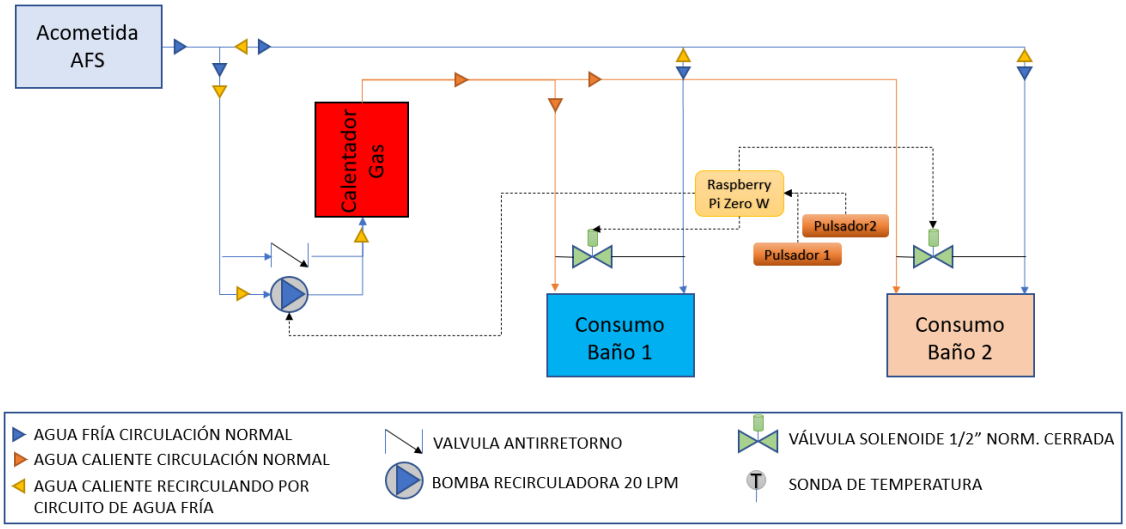

Figura 3.2 Control Centralizado

La gran ventaja de este sistema es que la ampliación de puntos de consumo equipados con recirculación es mucho más sencilla, ya que tan solo es necesario añadir una válvula de bypass en cada uno de ellos.

La desventaja de este sistema es la necesidad de realizar cableado entre los diferentes puntos, lo cual, no siempre es sencillo, y que, para implementar cualquier elemento adicional, como puede ser un sensor de temperatura o un botón adicional en cada baño, necesitará también realizar el consiguiente cableado.

Debido a esto, en este sistema, para evitar introducir agua caliente en el circuito de agua fría, se tendría que programar el funcionamiento del sistema mediante una consigna de tiempo en cada baño. Ese tiempo se obtendría de manera empírica en cada uno de los puntos de consumo.

Igualmente sería posible realizar este sistema con la placa Raspberry Pi Pico.

3.2.3. Sistema descentralizado

Se trata del sistema más versátil, dado que disponemos de una controladora en cada uno de los baños.

Esto nos permite disponer de los elementos que nos hagan falta en cada uno de los puntos de consumo, y en la generación de ACS, sin necesidad de realizar cableado entre estancias.

En esta configuración, tenemos la posibilidad de disponer de sensores de temperatura en cada baño, con lo cual, siempre es posible detener la circulación de agua, en el momento en que llega a la temperatura deseada, sin introducirla (al menos de forma que no alcance una temperatura excesiva) en el circuito de agua fría.

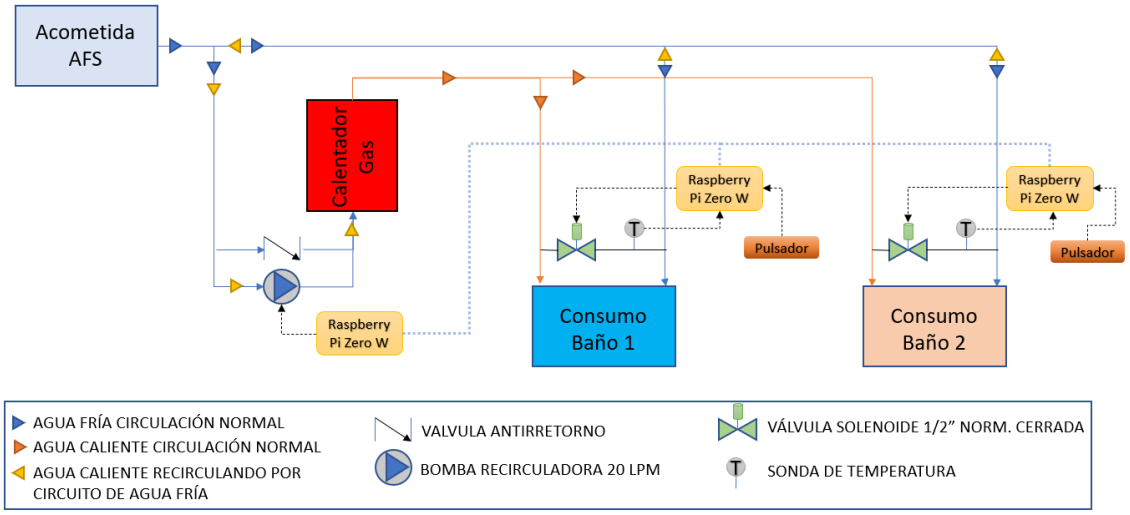

Figura 3.3 Sistema descentralizado

La principal ventaja de este sistema es, como hemos dicho, la versatilidad. Además de permitir la instalación de sensores en los baños, leds o altavoces de confirmación, etc.

La realización del presente trabajo se centra en la programación de los sistemas mencionados en los dos últimos puntos. El sistema centralizado y el descentralizado.

3.3. ELEMENTOS NECESARIOS PARA LA REALIZACIÓN DEL TRABAJO

3.3.1. Raspberry Pi Zero WH

Como se ha comentado anteriormente, se trata del modelo de la compañía con menores prestaciones, pero con sistema operativo (capaz de ejecutar varios procesos al mismo tiempo), y con comunicación WIFI, para poder desarrollar el sistema descentralizado. También dispone de los conectores soldados, para facilitar el conexionado sin soldaduras.

Figura 3.4 Raspberry Pi Zero WH

3.3.2. Bomba Recirculadora

Se calcula que una bomba de 10 lpm y 5 mca, será suficiente para que haga llegar el agua caliente en los siguientes tiempos para cada baño:

- Baño más alejado (Baño 1) → 45 segundos, dado que es necesario circular 7,5 litros para que llegue el agua caliente
- Baño más cercano (Baño 2) → 30 segundos, dado que es necesario circular 5 litros para que llegue el agua caliente

Por tanto, una bomba recirculadora tal y como la que se muestra a continuación, sería suficiente para la aplicación, ya que, para las condiciones de partida indicadas anteriormente, nos daría un caudal y altura ligeramente superiores (11,15 lpm y 6,217 mca).

Figura 3.5 Imagen Bomba (Grundfos) Modelo ALPHA1 L 25-60 180

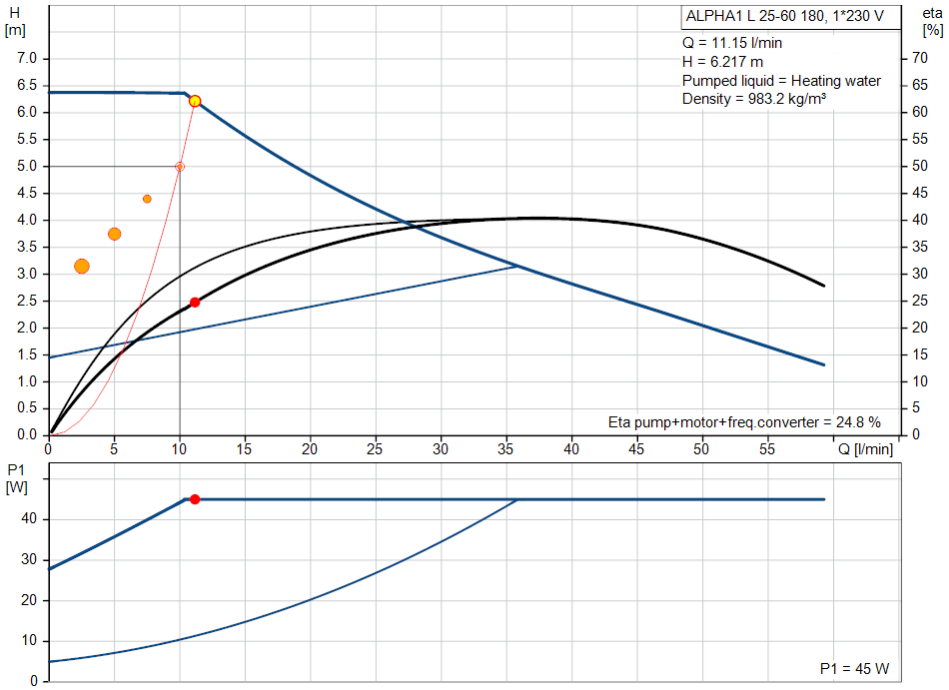

Figura 3.6 Curva de trabajo y Punto de Trabajo Bomba (Grundfos) ALPHA1 L 25-60 180

Se trata de un equipo de primera marca, por lo que obviamente, su precio es alto (sobre 180€). Es posible encontrar equipos de gama de entrada por un precio alrededor de un 50% inferior a este.

3.3.3. Válvula Solenoide

En cada uno de los baños, debemos disponer una válvula solenoide que nos comunique los circuitos de agua caliente y fría, y que por tanto nos permita realizar la recirculación de agua deseada.

Estas válvulas estarán situadas bajo el lavabo del cada cuarto húmedo, dado que, en ese punto, las tuberías están completamente accesibles, y estos elementos quedarán ocultos tras el mueble de lavabo (si lo hay) o bien será poco visible.

Estos solenoides se seleccionan NC (Normalmente Cerrada) para que solamente sea necesario energizarlas en el periodo de calentamiento del circuito, siendo este menor de 1 minuto.

Las válvulas serán DN15 (1/2”), al igual que las tomas del lavabo.

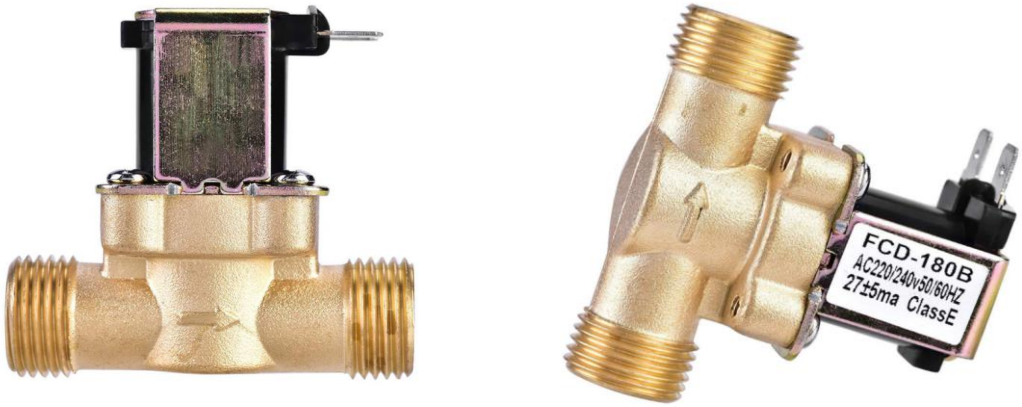

Figura 3.7 Válvula Solenoide ½" Normalmente NC (Normalmente Cerrada)

3.3.4. Válvula Antirretorno

La válvula antirretorno, es necesaria para evitar que la circulación de agua para calentamiento haga bypass con el circuito normal de alimentación de agua fría a la caldera.

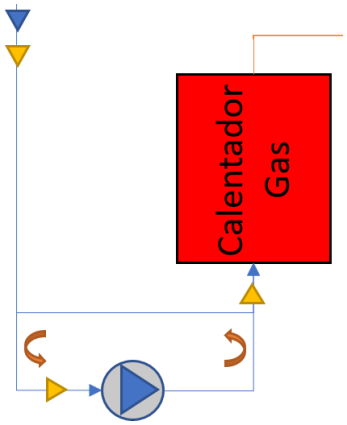

Figura 3.8 Funcionamiento sin válvula antirretorno

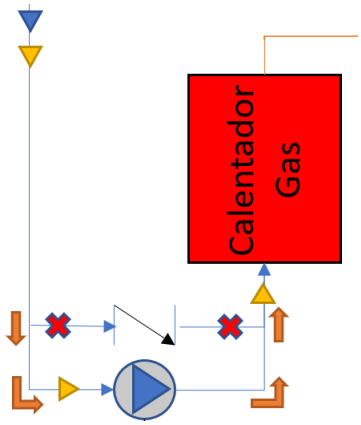

Figura 3.9 Funcionamiento con válvula antirretorno

Por este motivo, se colocará la válvula antirretorno en este punto del circuito. Esta válvula, también puede sustituirse por una solenoide de 1" que esté normalmente abierta (NO), y se cierre durante el funcionamiento de la bomba circuladora.

Figura 3.10 Válvula Antirretorno Genebre

3.3.5. Relé con opto acoplamiento

No es posible abrir o cerrar el paso de corriente a cada uno de los elementos mediante la Raspberry Pi, ya que los GPIO tan solo proporcionan 3,3V y una corriente máxima de 16mA.

El voltaje y la potencia ofrecida por la placa, no es suficiente para alimentar a cada uno de los elementos.

Tampoco es posible realizar la operación mediante relés directamente, ya que cortocircuitaríamos el GPIO de la RPi quemando la placa.

Por este motivo, se utilizarán Relés con opto acoplamiento, que nos permiten controlar el contacto solo con la tensión de 3,3V del GPIO sin quemar la placa, ya que este sistema separa físicamente la señal de activación del circuito.

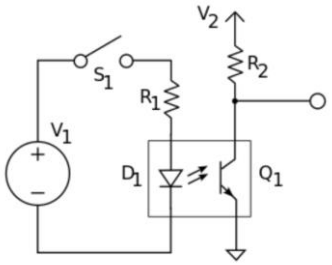

Figura 3.11 Esquema Optoacoplador

El opto acoplamiento se realiza mediante un LED y un fototransistor. La única comunicación entre ambos circuitos es la luz emitida por el diodo.

Todos los elementos utilizados en este trabajo son de alimentación en 230V en CA. Esto nos evita tener que disponer de fuentes de alimentación adicionales en las diferentes estancias.

Dado que se va a contemplar la instalación de relés en 3 lugares diferentes, se ha elegido la colocación de módulos de un único relé opto acoplado. Existen en el mercado módulos de 2, 4, 6, …

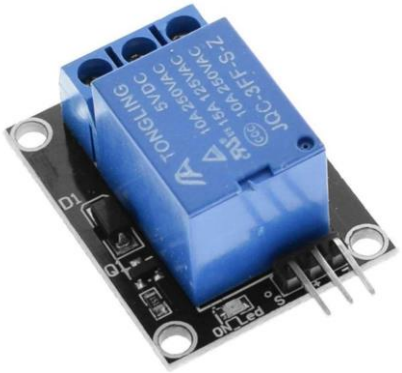

Figura 3.12 Modulo de 1 Canal de Relé Opto acoplado

3.3.6. Sonda de temperatura

Se utiliza para la medición de la temperatura en el punto de bypass. Servirá para desconectar la recirculación cuando el agua caliente llegue a dicho punto.

Figura 3.13 Sonda de temperatura DS18B20

Este sensor de temperatura (DS18B20) utiliza el protocolo de comunicación 1-Wire, lo que nos permite una fácil comunicación con él, y la posibilidad de crear un red de sensores, sin tener que ocupar muchos de nuestros pines de entradas y salidas de datos de nuestra Raspberry Pi.

3.3.7. Otros elementos

Placa de prototipado

Se utiliza para conexionar los diferentes elementos de forma provisional para realizar pruebas.

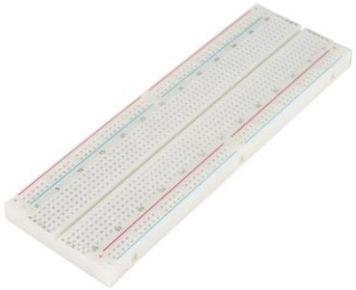

Figura 3.14 Placa de Prototipado

Cables Dupont

Se utilizan para las conexiones de los diferentes elementos (no los de potencia). Tienen la ventaja de facilitar la conexión entre los pines de la RPi, los de los Relés y los terminales de la placa de prototipado. Existen cables H-H

(Hembra – Hembra), M-M (Macho – Macho) y M-H (Macho – Hembra). También se denominan Jumpers.

Figura 3.15 Cables DuPont

3.4. INICIACIÓN CON RASPBERRY PI

3.4.1. Instalación del sistema operativo

Como se ha indicado anteriormente, salvo el modelo Pico, las RPi funcionan con un sistema operativo específico llamado Raspberry Pi OS (Anteriormente Raspbian).

Para su instalación, se puede descargar la aplicación Raspberry Pi Imager para Windows, Mac o Linux de la página web (https://www.raspberrypi.org/software/)

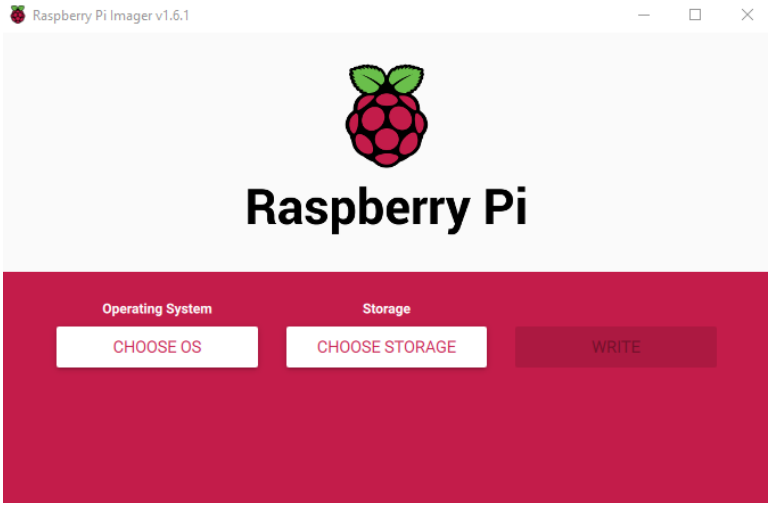

Figura 3.16 Raspberry Pi Imager v.1.6.1

Tras esto se escoge como sistema operativo Raspberry Pi OS (32-bit) y como almacenamiento, una tarjeta Micro SD que será la que posteriormente se utilizará como almacenamiento en la RPi.

Figura 3.17 Proceso Instalación Raspberry Pi OS

A continuación, se muestra el conexionado de la placa para su alimentación y su conexión a los periféricos de visualización y control.

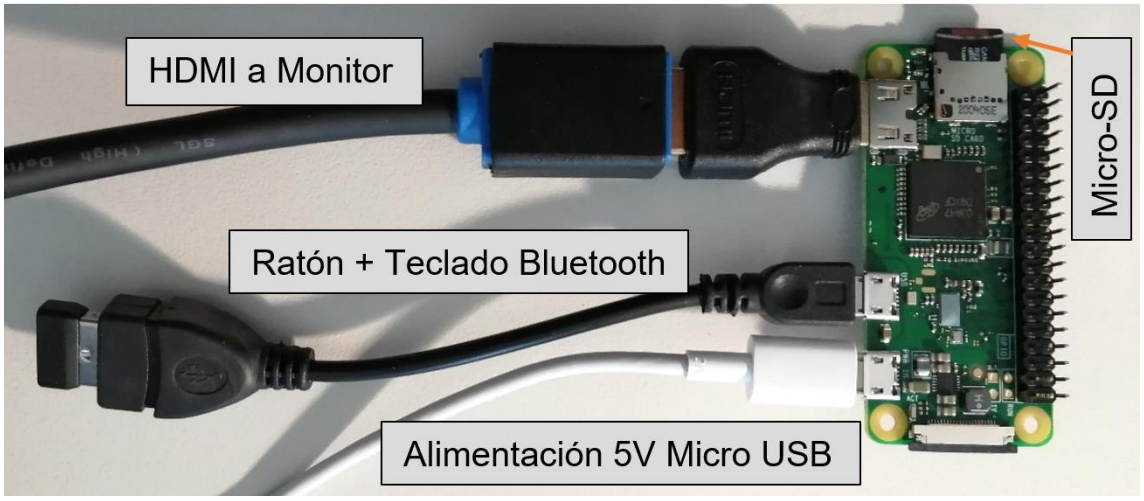

Figura 3.18 Conexionado Alimentación y Periféricos

3.4.2. Conexión y Configuración Wifi

Tras conectar la RPi con un monitor, ratón y teclado, se procede a configurar la conexión Wifi, utilizando la interfaz del sistema operativo.

Figura 3.19 Conexión a red Wifi

Tras esto, y con el fin de poder acceder siempre al equipo mediante una IP estática (que no cambie cada vez que reiniciemos el equipo), se procede a editar el archivo dhcpcd.conf, existente en la ruta /etc/dhcpcd.conf de la RPi. Este proceso se realiza mediante la ventana de comandos de la RPi.

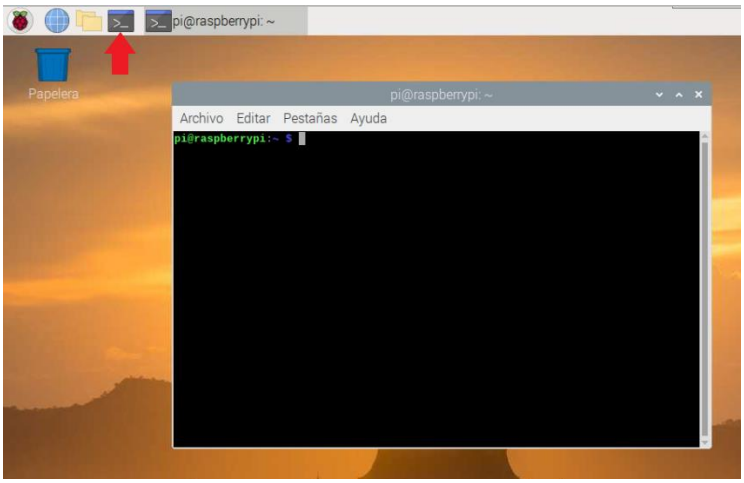

Figura 3.20 Ventana de Comandos RPi

Se introduce la siguiente instrucción para acceder al archivo de configuración:

```
sudo nano /etc/dhcpcd.conf
```

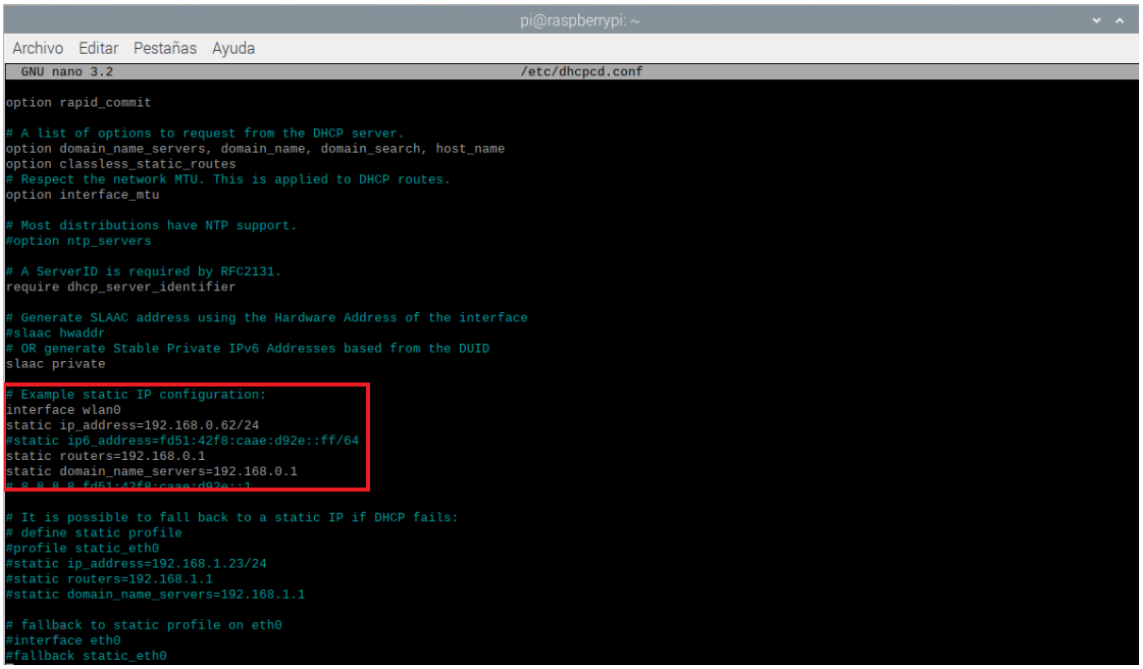

Figura 3.21 Configuración IP Estática. Edición de fichero "dhcpcd.conf"

Editamos el fichero, lo guardamos y reiniciamos la RPi para disponer de dicha configuración.

3.4.3. Acceso remoto a través de VNC

La RPi, dispone preinstalado de un software de acceso remoto denominado VNC Server. Este nos permite acceder a la RPi desde un dispositivo externo, siempre que se disponga del software instalado (VNC Client), con tan solo introducir la IP, usuario y contraseña.

En la RPi simplemente hay que activar el servicio en el apartado Configuración de Raspberry Pi, en la pestaña Interfaces.

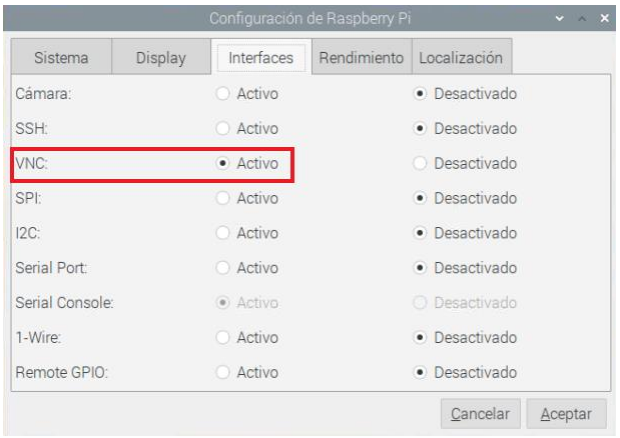

Figura 3.22 Activación de VNC Server

3.4.4. Programación de RPi con Python

Raspberry Pi dispone de un editor Python llamado Thonny Python (aunque es posible instalar otros editores).

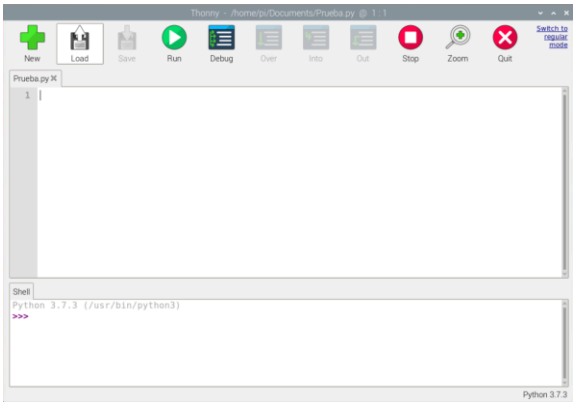

Figura 3.23 Interfaz Thonny Python

Tras editar y guardar un archivo en la tarjeta de memoria de la RPi, se puede ejecutar el programa, bien utilizando el editor, presionando en “Run”, que equivale a las siguientes líneas de código:

```
cd /home/Pi/Documents/
```

Accedemos al directorio en el que está el archivo que estamos editando.

```
run Prueba.py
```

Ejecutamos el programa contenido en el archivo “Prueba.py” (en este caso).

Figura 3.24 Ejecución de programas en Thonny Python

La otra opción es la ejecución del programa desde la ventana de comandos de la RPi con el siguiente código:

cd Documents

Accedemos al directorio donde se encuentra el archivo de código.

Sudo python3 Prueba.py

Ejecutamos el archivo “Prueba.py” con Python3.

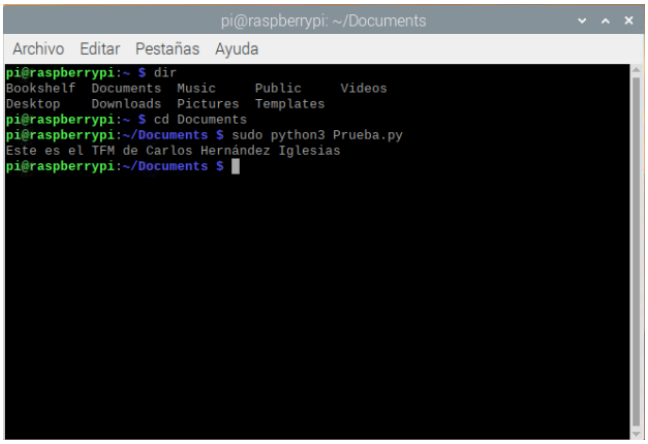

Figura 3.25 Ejecución de archivo Python en la consola de la RPi

3.5. PROGRAMACIÓN GPIO EN RASPBERRY PI

Una de las utilidades de que dispone RPi, es la existencia de terminales GPIO (General Purpose Input Output), los cuales se pueden utilizar para accionar elementos, y para recibir datos de sensores y otros elementos.

Para poder gestionar estos terminales se ha de disponer de la biblioteca RPi.GPIO, la cual viene preinstalada en la RPi, aunque también hay desarrolladas otras.

A continuación, se recuerda el esquema de conexiones de que dispone RPi.

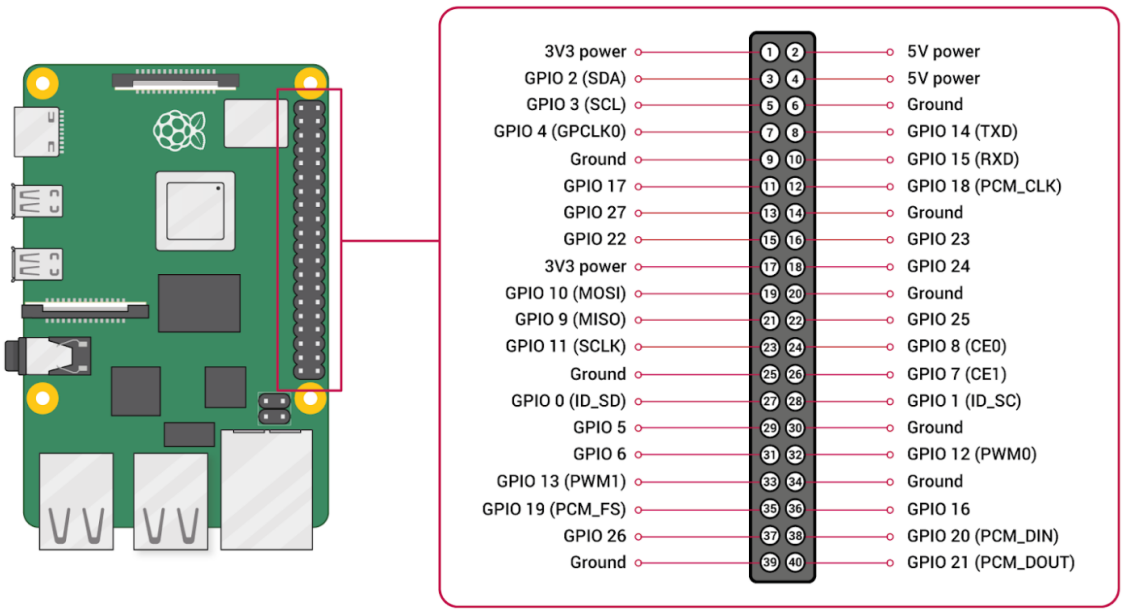

Figura 3.26 Esquema GPIO Rasberry Pi (aprendiendoarduino.wordpress.com)

Para mostrar la forma de programar GPIO en una RPi, a continuación, se indica el procedimiento para realizar una aplicación de muestra en la que al pulsar un botón (Entrada), se inicia una secuencia de encendido de LEDs (Salidas).

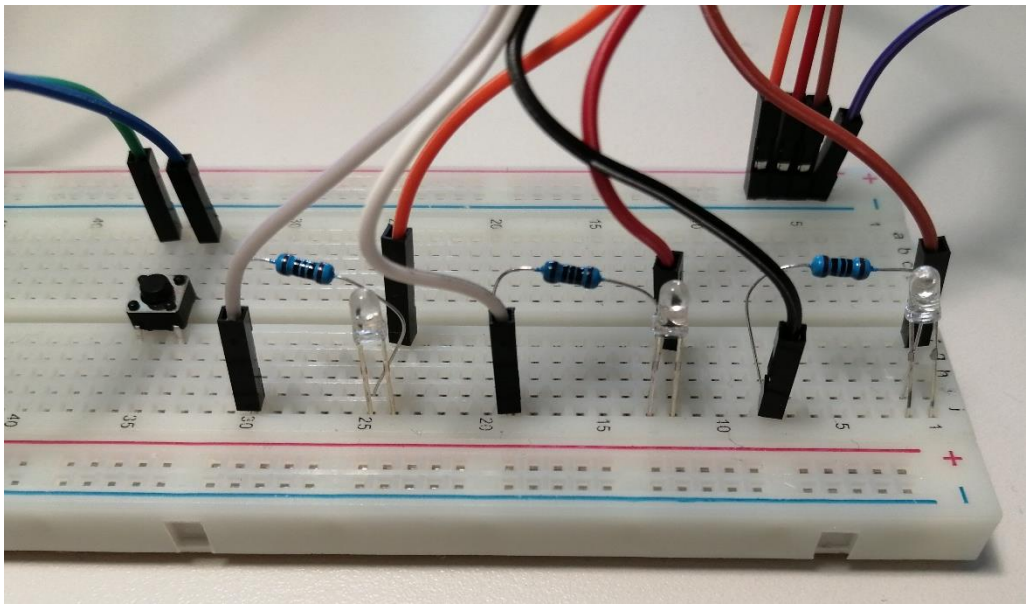

Figura 3.27 Aplicación GPIO con LEDs

Entre cada salida GPIO y cada led, se pone una resistencia que nos limita la corriente que pasa a través de la placa (Máximo 60mA por GPIO), para que no se dañe.

A continuación, se indica y explica el código introducido para este ejercicio:

```
import time
```

Se importa la librería time

```
import RPi.GPIO as gpio
```

Se importa la librería RPi.GPIO, y se genera el objeto gpio

```
gpio.setwarnings(False)
```

Se desactivan los avisos de error de la librería.

```
gpio.setmode(gpio.BOARD)
```

Definimos la numeración de los contactos que vamos a utilizar. Existe la numeración de la propia placa (Board, la que utilizamos en este caso) y la nomenclatura que le da a los contactos RPi (BCM). (Ver figura 3.25).

```
gpio.setup(3, gpio.OUT)
gpio.setup(5, gpio.OUT)
gpio.setup(7, gpio.OUT)
gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)
```

Indicamos la numeración de los GPIO de la RPi que vamos a utilizar, definiendo los que van a ser salidas (gpio.OUT, en este caso el 3, 5 y 7) y los que van a ser entradas (gpio.IN, el número 11).

```
while True:
    boton=gpio.input(11)
```

Creamos una variable que recoge el estado del botón. Cuando el botón está presionado, la variable valdrá 0,y cuando esté libre, 1.

```
    if boton==1:
        gpio.output(3,gpio.LOW)
        gpio.output(5,gpio.LOW)
        gpio.output(7,gpio.LOW)
```

Si el botón no está apretado (al llegar a este punto del código), las salidas de los LED estarán en reposo y, por tanto, estos, apagados.

```
    else:
        for i in range(10):
            gpio.output(3,gpio.HIGH)
            gpio.output(5,gpio.LOW)
            gpio.output(7,gpio.LOW)
            time.sleep(0.2)
            gpio.output(3,gpio.LOW)
            gpio.output(5,gpio.HIGH)
            gpio.output(7,gpio.LOW)
            time.sleep(0.2)
            gpio.output(3,gpio.LOW)
            gpio.output(5,gpio.LOW)
            gpio.output(7,gpio.HIGH)
            time.sleep(0.2)
```

#Cuando el botón se acciones, el programa realizará 10 veces la secuencia de encender cada uno de los leds secuencialmente (con una pausa de 0,2 segundos), para volver a apagarse de nuevo (si el botón no está presionado al terminar la secuencia).

Para evitar estar comprobando continuamente el estado del botón, existe una función de la librería RPi.GPIO llamada “wait_for_edge”, que para el programa quedando a la espera de que suceda lo que indiquemos:

- FALLING → Que pase a valer “0”, o lo que es lo mismo, que presionemos el botón. Nuestro caso
- RISING → Que pase a valer “1”, o lo que es lo mismo, que soltemos el botón.

El código quedaría de la siguiente forma:

```
import time
import RPi.GPIO as gpio
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD)
gpio.setup(3, gpio.OUT)
gpio.setup(5, gpio.OUT)
gpio.setup(7, gpio.OUT)
gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)

gpio.output(3,gpio.LOW)
gpio.output(5,gpio.LOW)
gpio.output(7,gpio.LOW)
# Establecemos como estado inicial de los LED como apagados
while True:
gpio.wait_for_edge(11,gpio.FALLING)
# El programa queda a la espera de que presionemos el botón
for i in range(10):
    gpio.output(3,gpio.HIGH)
    gpio.output(5,gpio.LOW)
    gpio.output(7,gpio.LOW)
    time.sleep(0.2)
    gpio.output(3,gpio.LOW)
    gpio.output(5,gpio.HIGH)
    gpio.output(7,gpio.LOW)
    time.sleep(0.2)
```

gpio.output(3,gpio.LOW)

gpio.output(5,gpio.LOW)

gpio.output(7,gpio.HIGH)

time.sleep(0.2)

Ejecutamos la secuencia en 10 ocasiones

gpio.output(3,gpio.LOW)

gpio.output(5,gpio.LOW)

gpio.output(7,gpio.LOW)

Devolvemos los LED a su estado original (apagado)

De esta forma, simplificamos mucho el código, además de evitar estar comprobando continuamente el estado de la variable.

3.6. COMUNICACIÓN ENTRE EQUIPOS RASPBERRY PI VÍA WEB

Uno de los objetivos del presente trabajo, pasa por conseguir la comunicación entre las diferentes placas, de forma que estén coordinadas para realizar las diferentes tareas que se les demandan.

Existen diferentes formas de realizar esta comunicación, aunque en este trabajo se ha optado por realizarlo mediante comunicación web.

Existen webs estáticas y dinámicas. Las estáticas están alojadas en un servidor y siempre muestran la misma información, y en cambio las dinámicas nos permiten generar de forma activa la página cada vez que se realiza una solicitud. Este último formato, nos permite dos cosas:

- Almacenar datos en páginas web.
- Solicitar datos a una web que disponga de estos datos
- Realizar solicitudes de ejecución de código en la/s placa/s que realicen la función de servidor.

3.6.1. Programación de un servidor web en RPi

Como primer paso, necesitamos disponer de un servidor web en al menos una de las placas, y para esto, necesitamos disponer en esta de un software específico para ello. En nuestro caso LAMP.

Este software está compuesto por los siguientes componentes:

L: Linux

A: Apache

M: MySQL

P: PHP

Linux sirve como sistema operativo base para ejecutar el servidor web Apache. Como Apache no sabe interpretar contenidos dinámicos, este le envía un código fuente al intérprete PHP, incluyendo la información correspondiente sobre las acciones del visitante de la web, y permite el acceso a la base de datos MySQL. El resultado es devuelto a Apache y este se muestra finalmente en el navegador web del visitante. (IONOS)

En nuestro caso, tan solo necesitamos utilizar HTML, por lo que no necesitamos instalar PHP, ni tampoco MySQL para funcionar con bases de datos.

Para instalar apache, se escribe en la consola:

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

Una vez instalado, introduciendo la IP de la placa en la que se ha instalado el servidor (desde cualquier equipo con acceso a la red LAN), podemos comprobar que funciona correctamente.

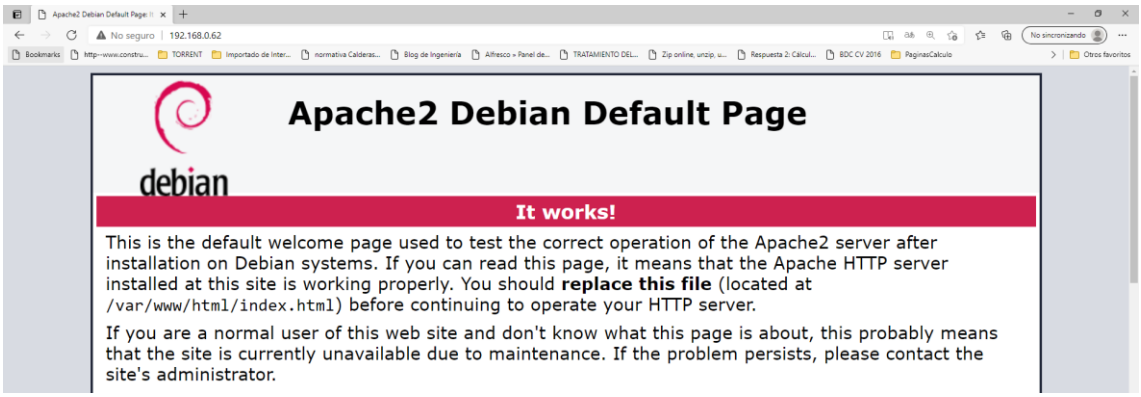

Figura 3.28 Página Inicial Apache2 Debian

Llegados a este punto, ya disponemos de un servidor HTML en la placa.

A continuación, vamos a programar un ejemplo de web simple como ejemplo de programación web en Python.

Para hacerlo posible se ha elegido la librería Flask de Python. Existe la posibilidad de realizar la programación generando la web por completo cada vez que hay una solicitud, o bien cargando una plantilla y modificando las partes que necesitamos. En nuestro caso por sencillez, vamos a ver cómo se realiza una web muy sencilla sin necesidad de generar una plantilla.

```
from flask import Flask
```

Importamos la librería Flask, generando un objeto Flask.

```
app = Flask(__name__)

@app.route('/INICIO')
```

Definimos la ruta en que estará alojada la web. En este caso será la dirección IP del servidor seguida de /INICIO)

```
def INICIO():
```

```
    return "Este es el TFM de Carlos Hernández Iglesias"
```

Cuando entremos en dicha dirección, la Web nos devolverá el texto entre comillas.

```
if __name__=="__main__":
    app.run(host='0.0.0.0', port=60, debug=True)
```

Definimos la ubicación del servidor (el propio equipo), el puerto de acceso (60) y si queremos depurar.

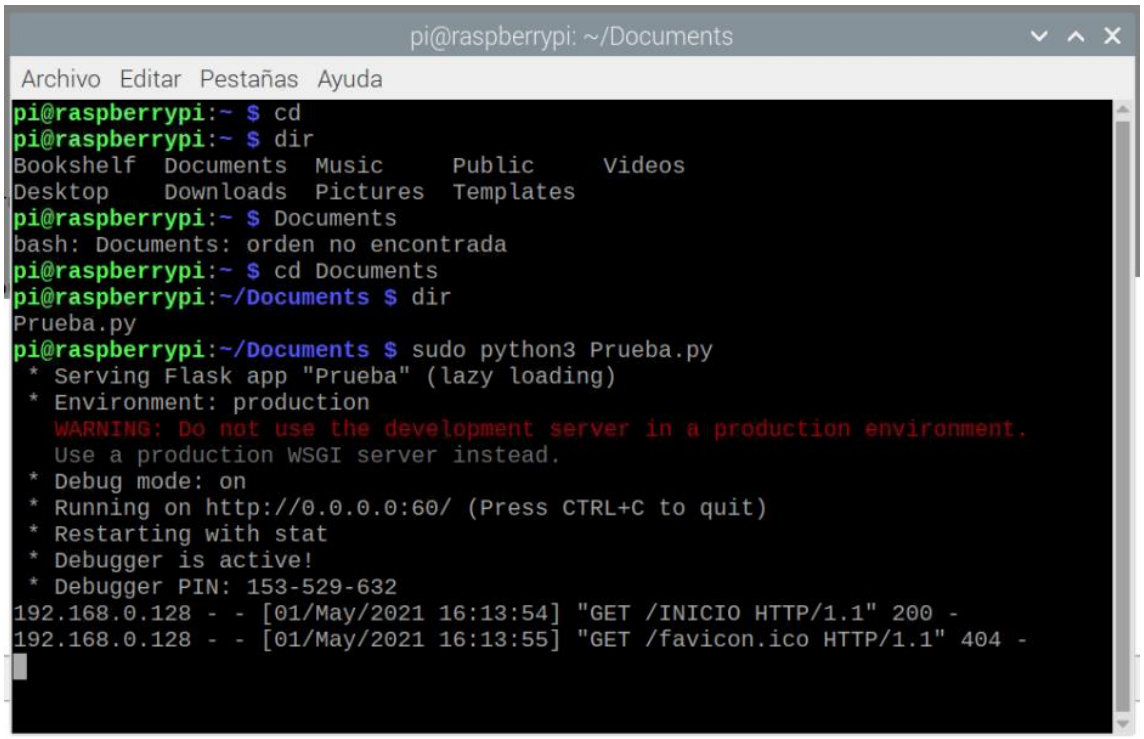

Figura 3.29 Ejecución Servidor Web

Desde un PC conectado a la misma red, podemos comprobar al introducir la dirección de la web, como aparece el texto introducido en el programa.

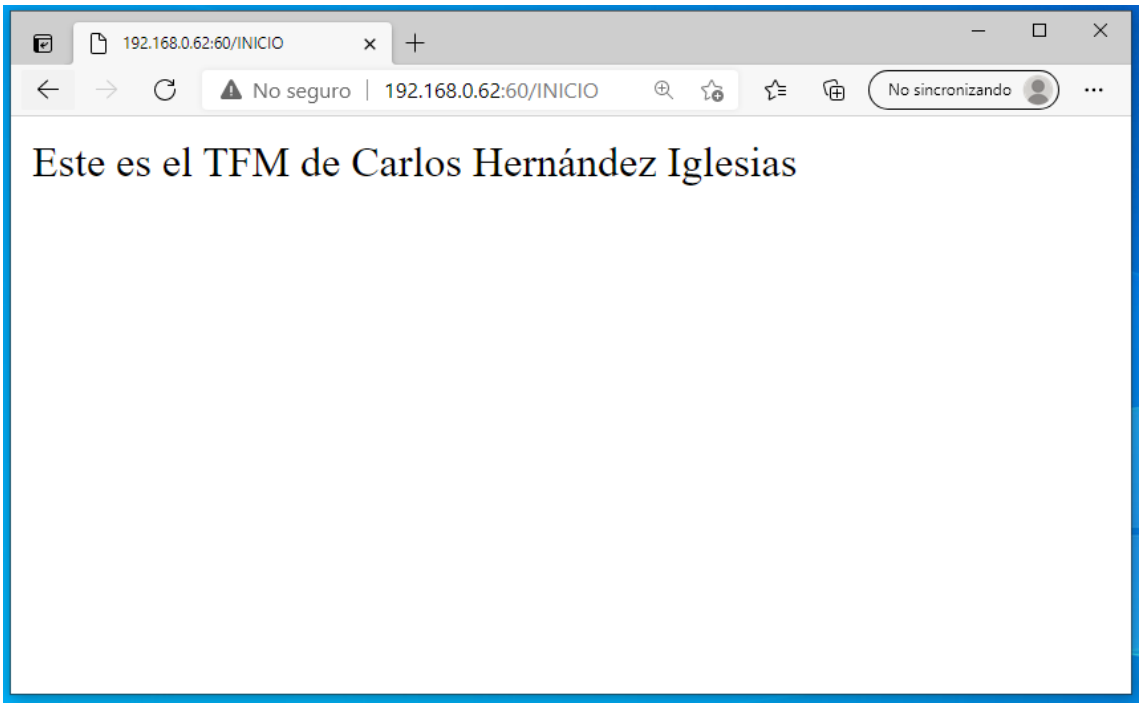

Figura 3.30 Web Hospedada en Raspberry Pi

3.6.2. Realización de consultas web desde Raspberry Pi

Para poder consultar un dato de una web, ya sea interna (hospedada en un equipo dentro de una LAN particular) o bien externa, utilizaremos la librería de Python “urllib”. Para ilustrar la utilidad de esta librería, consultaremos los datos de la página creada en el apartado anterior, desde otra placa RPi.

```
import urllib.request
```

Importamos la librería urllib

```
respuesta=urllib.request.urlopen('http://192.168.0.62:60/INICIO')
```

Creamos la variable respuesta, que contiene la web que hemos solicitado

```
page=respuesta.read()
```

Creamos la variable page, que contiene los caracteres de la web

```
texto=page.decode()
```

Creamos la variable texto, que contiene el texto de la web (ya convertido a texto). En caso de estar consultando un número, deberemos convertirlo a integer con la función int().

```
print(texto)
```

Imprimimos en la consola el texto extraído de la web.

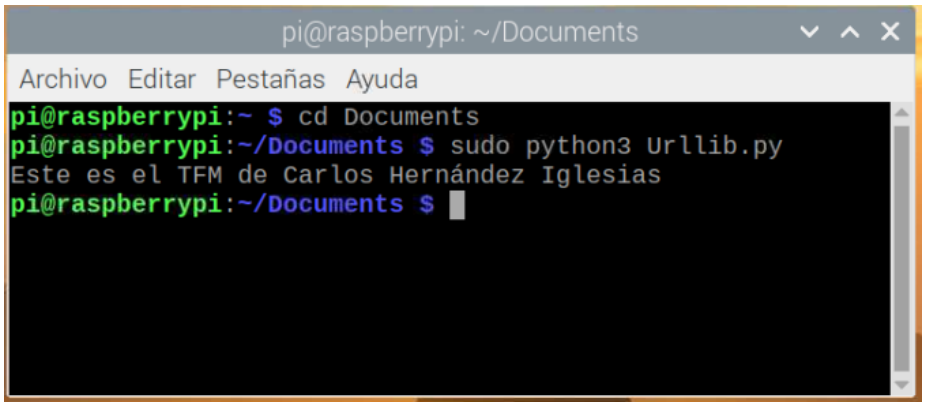

Figura 3.31 Ejecución Consulta Web

4. IMPLEMENTACIÓN DE UN SISTEMA DE RECIRCULACIÓN DE ACS SIN TUBERÍA DE RETORNO MEDIANTE EQUIPOS RASPBERRY PI

4.1. IMPLEMENTACIÓN MEDIANTE UNA PLACA Y CABLEADO

4.1.1. Configuración

La implementación de este sistema es muy sencilla, ya que tan solo necesitamos:

- Disponer una placa RPi en la que utilizaremos:
  - 2 Entradas
    - Solicitud de calentamiento Baño 1 mediante botón
    - Solicitud de calentamiento Baño 2 mediante botón
  - 3 Salidas
    - Activación de la bomba
    - Apertura de la válvula bypass del baño 1
    - Apertura de la válvula bypass del baño 2
- Medir el tiempo que tarda en llegar el agua caliente al punto de bypass desde que accionamos la bomba.

La placa se podría colocar en el punto desde el que fuera más sencillo instalar a posteriori el cableado. Este podría ser el cuadro general de la vivienda, ya que de él parten todas las canalizaciones a las diferentes estancias. Se optaría por alojar allí tanto la placa como los 3 relés y los 2 botones y se llevaría en 230V el accionamiento de los equipos, para evitar posibles problemas de caída de tensión en corriente continua debido a las distancias.

Se asignarían las entradas y salidas de la placa de la siguiente forma:

- Solicitud de calentamiento Baño 1 mediante botón → GPIO 11
- Solicitud de calentamiento Baño 2 mediante botón → GPIO 16
- Activación de la bomba → GPIO 3
- Apertura de la válvula bypass del baño 1 → GPIO 5
- Apertura de la válvula bypass del baño 2 → GPIO 7

4.1.2. Programación

El código necesario para el funcionamiento sería el siguiente:

```
import time

import RPi.GPIO as gpio

gpio.setwarnings(False)
```

Importamos las librerías time y RPi.GPIO, para poder disponer de tiempos de espera en el código y para poder controlar los terminales de entradas y salidas.

```
gpio.setmode(gpio.BOARD)
```

Utilizamos la numeración de la placa en lugar de la propia de RPi.

```
gpio.setup(3, gpio.OUT)
gpio.setup(5, gpio.OUT)
gpio.setup(7, gpio.OUT)
gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)
gpio.setup(16, gpio.IN,pull_up_down=gpio.PUD_UP)
```

Definimos 3 GPIO como salidas para los relés (GPIO 3, 5 y 7) y como entradas los 2 botones (GPIO 11 y 16).

```
while True:
    boton1=gpio.input(11)
    boton2=gpio.input(16)
```

Generamos dos variables que almacenan en cada iteración del bucle el estado de ambos botones (‘0’ pulsado, ‘1’ sin presionar).

```
    if boton1==1 and boton2==1:
        gpio.output(3,gpio.LOW)
        gpio.output(5,gpio.LOW)
        gpio.output(7,gpio.LOW)
```

Si ambos botones no están presionados, se mantienen las salidas sin actuación.

```
    else:
        if boton1==0
            gpio.output(3,gpio.HIGH)
            gpio.output(5,gpio.HIGH)
```

Si está presionado el botón 1, activamos las salidas correspondientes a la electroválvula del baño 1 y la bomba recirculadora.

```
            time.sleep(150)
            gpio.output(3,gpio.LOW)
            gpio.output(5,gpio.LOW)
```

Esperamos el tiempo que previamente hemos medido que tarda la bomba en llevar agua caliente a ese baño y desconectamos ambos equipos (electroválvula y bomba).

```
else
    gpio.output(3,gpio.HIGH)
    gpio.output(7,gpio.HIGH)
    time.sleep(80)
    gpio.output(3,gpio.LOW)
    gpio.output(7,gpio.LOW)
```

En caso de que sea el botón 2, realizamos la misma operación, pero en lugar de la electroválvula del baño 1, lo realizamos con la del baño 2.

Como se puede observar, se trata de una programación muy simple, que tan solo requiere el uso de los GPIO de la placa y la librería tiempo para poder realizar las esperas hasta que llegue el agua caliente.

Este sistema tiene el inconveniente de que si al solicitar agua caliente para alguno de los baños, se ha solicitado anteriormente para otro baño, podemos introducir gran cantidad de agua caliente en el circuito de agua fría, y, por tanto, a la hora de utilizar el grifo, al principio solo saldría agua caliente, tanto con el mando en caliente, como en fría.

4.1.3. Estimación del coste

A continuación, se resumen los costes principales de este sistema.

Item	Descripción	Ud	P.Unit	Total
1	SISTEMA CENTRALIZADO	1	206,62 €	206,62 €
1.01	Raspberry Pi Zero WH	1	15,95 €	15,95 €
1.02	Fuente de Alimentación 5V 2A Micro-USB	1	6,10 €	6,10 €
1.03	Tarjeta MicroSDHC 16GB CLASS10	1	5,01 €	5,01 €
1.04	Relé Optoacoplado 10A 230V	3	1,99 €	5,97 €
1.05	Bomba Circuladora 10 lpm 5mca	1	109,00 €	109,00 €
1.06	Válvula Solenoide 3/4" 220V	2	11,99 €	23,98 €
1.07	Válvula Antirretorno 1"	1	9,01 €	9,01 €
1.08	Cable unipolar H07V-K 1,5mm2	120	0,18 €	21,60 €
1.09	Pequeño Material	1	10,00 €	10,00 €

Tabla 4.1 Coste Sistema Centralizado

Como principal diferencia con respecto a los sistemas que veremos más adelante, se puede ver el cableado a los diferentes elementos. Este punto está valorado para un cableado de 2 hilos de 1,5 mm^2 a una distancia de 20 metros (medida sobre la canalización existente en la vivienda). Este coste no tiene en cuenta la mano de obra de la colocación de dichos cables, lo cual aumentaría el coste aproximadamente 35 euros.

4.2. IMPLEMENTACIÓN MEDIANTE TRES PLACAS CON COMUNICACIÓN VIA WEB

4.2.1. Configuración

La implementación de este sistema es algo más complejo. La ventaja sobre el anterior es que no necesita s cableado por la vivienda, ya que situaremos cada placa en el punto donde necesitemos controlar un elemento:

- Disponer una placa RPi en la caldera, en la cual utilizaremos
  - Una salida para la activación de la bomba.
  - Un servidor Web que nos permitirá recibir instrucciones de las otras placas y dar información sobre el estado de funcionamiento del proceso.
- Disponer una placa RPi en cada uno de los cuartos húmedos que queremos recircular (en nuestro caso 2). En cada baño utilizaremos:
  - 1 Entrada
    - Solicitud de calentamiento Baño 1/2 mediante botón
  - 1 Salida
    - Apertura de la válvula bypass del baño 1/2
  - Solicitud Web para activación de la bomba
  - Consulta Web para conocer si el proceso está activo en el otro cuarto húmedo.

Al igual que en el caso anterior, necesitamos disponer de la medida del tiempo que tarda en llegar el agua caliente al punto de bypass desde que accionamos la bomba.

Se asignarían las entradas y salidas de las placas son de la siguiente forma:

- Caldera
  - Activación de la bomba → GPIO 16 (Placa 3)
- Baño 1
  - Solicitud de calentamiento Baño 1 (botón) → GPIO 11 (Placa 1)
  - Apertura de la válvula bypass del baño 1 → GPIO 16 (Placa 1)

- Baño 2
  - Solicitud de calentamiento Baño 2 (botón) → GPIO 11 (Placa 2)
  - Apertura de la válvula bypass del baño 2 → GPIO 16 (Placa 2)

La lógica de funcionamiento del sistema se muestra a continuación:

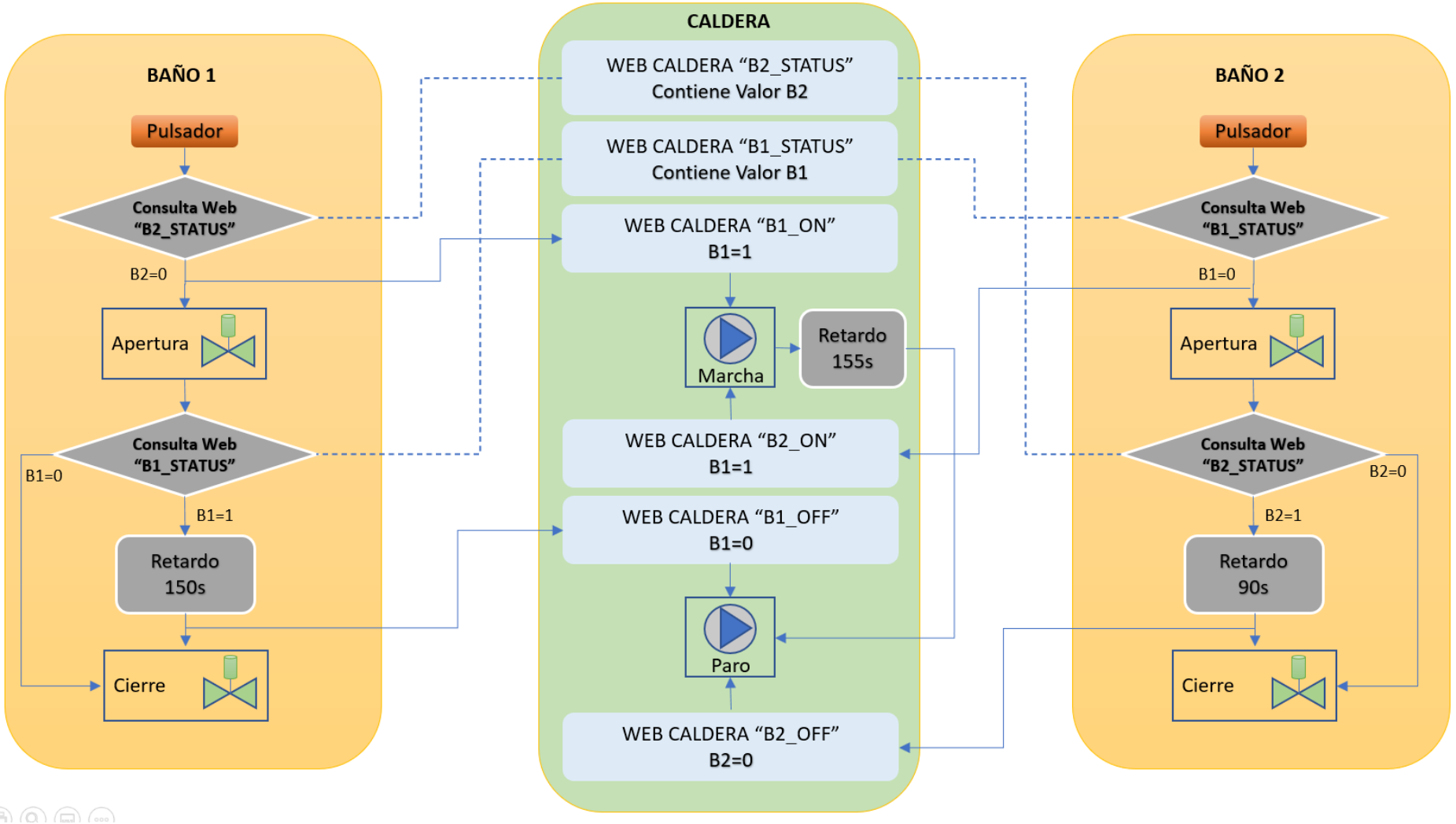

Figura 4.1 Lógica Funcionamiento con comunicación Web

Como se puede comprobar en el esquema anterior, el proceso sería el siguiente:

1. Las placas de los baños están a la espera de una pulsación, y la placa de la caldera, tiene un servidor web activo con varias webs.
2. Una vez se pulsa el botón de alguno de los baños, se realiza una consulta web para comprobar si el otro baño está recirculando. Si lo está haciendo acaba el proceso.
3. Si el otro baño no está recirculando, se abre la válvula de bypass y se envía una petición web para que arranque la bomba de recirculación.
4. Se comprueba que la petición web ha sido exitosa. En caso contrario se cierra la válvula y se termina el proceso (teniéndose que realizar de nuevo).
5. Si la bomba está en funcionamiento, se espera el tiempo prefijado para el baño en cuestión, y tras esto, se envía una petición web para la parada de la bomba y se cierra la válvula.
6. El proceso está terminado.

Para evitar que al final del proceso se pueda producir la situación de que la bomba quede recirculando con la válvula de bypass parada, se contempla una orden de parada en caso de no recibirse la petición web. Esto se realiza mediante una orden de parada fijada en un periodo superior al máximo entre el estipulado para ambos baños. En este caso superior a 150 segundos.

El código de cada una de las placas será el siguiente:

4.2.2. Programación Placa Caldera

Esta placa dispondrá del servidor web. Se omiten las importaciones de librerías.

```
from flask import Flask

import time

import RPi.GPIO as gpio

gpio.setwarnings(False)


gpio.setmode(gpio.BOARD)

gpio.setup(16, gpio.OUT)

gpio.output(16,gpio.LOW)

b1=0

b2=0
```

#Establecemos los valores de iniciales de la salida de la bomba, y de las variables correspondientes al funcionamiento de cada baño.

```
app = Flask(__name__)
```

```
@app.route('/B1_OFF')
def B1_OFF():
    gpio.output(16,gpio.LOW)
    global b1
    b1=0
    return "Bano 1 is OFF"
```

#Se sirve una web en la dirección (http://192.168.0.63/B1_OFF) que realiza lo siguiente al visitarla:

- Ordenamos la parada de la bomba
- Indicamos que vamos a utilizar la variable global del script “b1” (en caso contrario generaría una variable interna en esta función).
- Modificamos “b1” a “0” indicando que el proceso de calentamiento del baño 1 ha terminado.
- La web muestra “Bano 1 is OFF”

```
@app.route('/B1_ON')
def B1_ON():
    global b1
    b1=1
    gpio.output(16,gpio.HIGH)
    time.sleep(155)
    gpio.output(16,gpio.LOW)
    return "Bano 1 is ON"
```

#Se sirve una web en la dirección (http://192.168.0.63/B1_ON) que realiza lo siguiente al visitarla:

- Indicamos que vamos a utilizar la variable global del script “b1”
- Modificamos “b1” a “1” indicando que el proceso de calentamiento del baño 1 está en curso.
- Ordenamos el arranque de la bomba
- Esperamos 155 segundos y ordenamos la parada de la bomba, a modo de protección en caso de que esta no se haya parado por una orden anterior.
- La web muestra “Bano 1 is ON”

```
@app.route('/B1_STATUS')
def B1_STATUS():
    b1s=str(b1)
    return b1s
```

#Se sirve una web en la dirección (http://192.168.0.63:60/B1_STATUS) que realiza lo siguiente al visitarla:

- Recogemos “b1” y la pasamos a una variable tipo “string”.
- La web muestra el valor de “b1”

```
@app.route('/B2_OFF')
def B2_OFF():
    gpio.output(16,gpio.LOW)
    global b2
    b2=0
    return "Bano 2 is OFF"
```

#Se sirve una web en la dirección (http://192.168.0.63:60/B2_OFF) que realiza lo mismo que la equivalente a “B1_OFF” pero se refiere al baño 2:

```
@app.route('/B2_ON')
def B2_ON():
    global b2
    b2=1
    gpio.output(16,gpio.HIGH)
    time.sleep(95)
    gpio.output(16,gpio.LOW)
    return "Bano 2 is ON"
```

#Se sirve una web en la dirección (http://192.168.0.63:60/B2_ON) que realiza lo mismo que la equivalente a “B1_ON” pero se refiere al baño 2:

```
@app.route('/B2_STATUS')
def B2_STATUS():
    b2s=str(b2)
    return b2s
```

#Se sirve una web en la dirección (http://192.168.0.63:60/B2_STATUS) que realiza lo mismo que la equivalente a “B1_STATUS” pero almacena la variable b2:

```
if __name__=="__main__":
    app.run(host='0.0.0.0', port=60, debug=False)
```

#Se define el host y el puerto de acceso a estas webs

4.2.3. Programación Placa Baño 1/2

En las placas de los baños el script es idéntico. Solo se intercambian las referencias a las variables

```
import time
import urllib.request
import RPi.GPIO as gpio
gpio.setwarnings(False)
gpio.setmode(gpio.BOARD)
gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)
```

#Este pin lo utilizamos para registrar el accionamiento del botón.

```
gpio.setup(16, gpio.OUT)
gpio.output(16,gpio.LOW)
```

#Este pin lo utilizamos para activar el relé de la válvula. Se inicia en “0” para que la válvula esté cerrada.

```
while True:


    gpio.wait_for_edge(11,gpio.FALLING)
```

Se espera a que se presione el botón

```
    estado_b2=urllib.request.urlopen('http://192.168.0.63:60/B2_STATUS')
    b2=int(estado_b2.read())
```

Se consulta el estado del otro baño para comprobar si está realizando la recirculación (si no está en funcionamiento la variable, b2 en este caso, valdrá “0”)

```
    if b2==0:
        gpio.output(16,gpio.HIGH)
```

Abrimos la válvula de bypass

```
        t=urllib.request.urlopen('http://192.168.0.63:60/B1_ON')
```

Realizamos una petición web para que la RPi de la caldera arranque la bomba

```
        time.sleep(1)
        estado_b1=urllib.request.urlopen('http://192.168.0.63:60/B1_STATUS')
        b1=int(estado_b1.read())
```

Tras 1 segundo, realizamos una consulta web para comprobar que la RPi de la caldera ha recibido la orden de arranque de la bomba (si la ha recibido, la variable, en este caso b1, valdrá “1”)

```
if b1==1:
    time.sleep(150)
    t=urllib.request.urlopen('http://192.168.0.63:60/B1_OFF')
    gpio.output(16,gpio.LOW)
```

Si comprobamos que ha recibido la petición web, esperamos el tiempo estipulado para la llegada de agua caliente al cuarto húmedo y enviamos una petición web para la parada de la bomba. Tras esto se cierra la válvula de bypass y habríamos finalizado el proceso.

```
else:
    gpio.output(16,gpio.LOW)
    t=urllib.request.urlopen('http://192.168.0.63:60/B1_OFF')
```

Si no se hubiera recibido la petición web, cerramos la válvula de bypass y solicitamos vía web la parada de la bomba para asegurar que está apagada (aunque en principio no debería estar abierta, teniendo en cuenta que ya hemos comprobado que b1 no es igual a “1”.

4.2.4. Estimación del coste

A continuación, se resumen los costes principales de este sistema.

Item	Descripción	Ud	P.Unit	Total
2	SISTEMA CON COM. WEB	1	239,14 €	239,14 €
2.01	Raspberry Pi Zero WH	3	15,95 €	47,85 €
2.02	Fuente de Alimentación 5V 2A Micro-USB	3	6,10 €	18,30 €
2.03	Tarjeta MicroSDHC 16GB CLASS10	3	5,01 €	15,03 €
2.04	Relé Optoacoplado 10A 230V	3	1,99 €	5,97 €
2.05	Bomba Circuladora 10 lpm 5mca	1	109,00 €	109,00 €
2.06	Válvula Solenoide 3/4" 220V	2	11,99 €	23,98 €
2.07	Válvula Antirretorno 1"	1	9,01 €	9,01 €
2.08	Pequeño Material	1	10,00 €	10,00 €

Tabla 4.2 Coste Sistema con 3 Placas Comunicación Web

En este caso se incrementa el coste por la incorporación de más placas controladoras, pero eliminamos el coste del cableado.

4.3. UTILIZACIÓN DE SENSORES DE TEMPERATURA

4.3.1. Configuración

El problema de la utilización de tiempos como consigna de parada de la bomba y cierre de bypass es que, dependiendo del tiempo que haga que se ha accionado uno de los baños, el tiempo de recirculación del otro baño ha de ser

diferente y dependiente del tiempo que haga que se ha accionado el primer baño. Se pueden dar varios casos:

a) Al accionar el segundo baño, el agua de todo el circuito ya se ha enfriado: En este caso, los tiempos obtenidos empíricamente para cada baño son válidos y la solución planteada en el apartado anterior, es correcta.
b) Al accionar el segundo baño, el primero se encuentra recirculando: En este caso, podríamos programar el sistema para que esperara a que terminara el baño activo, y recirculara un tiempo inferior, que también tendríamos que haber medido (tiempo que tarda en llegar el agua caliente al segundo baño, cuando el primero ya dispone de ella).
c) Al accionar el segundo baño, el agua del primero ha comenzado a enfriarse: Este caso es el más complejo, ya que en función de lo que hagamos, puede que no lleguemos a la temperatura suficiente con la recirculación o bien que, si la temperatura es demasiado alta, se introduzca agua demasiado caliente en la tubería de agua fría y por tanto sea complicado conseguir una mezcla satisfactoria para el usuario.

Como se puede observar, el problema radica en el conocimiento de la temperatura del agua en el interior de las tuberías, por lo que, como mejora del sistema, se propone la instalación de sondas de temperatura en los baños.

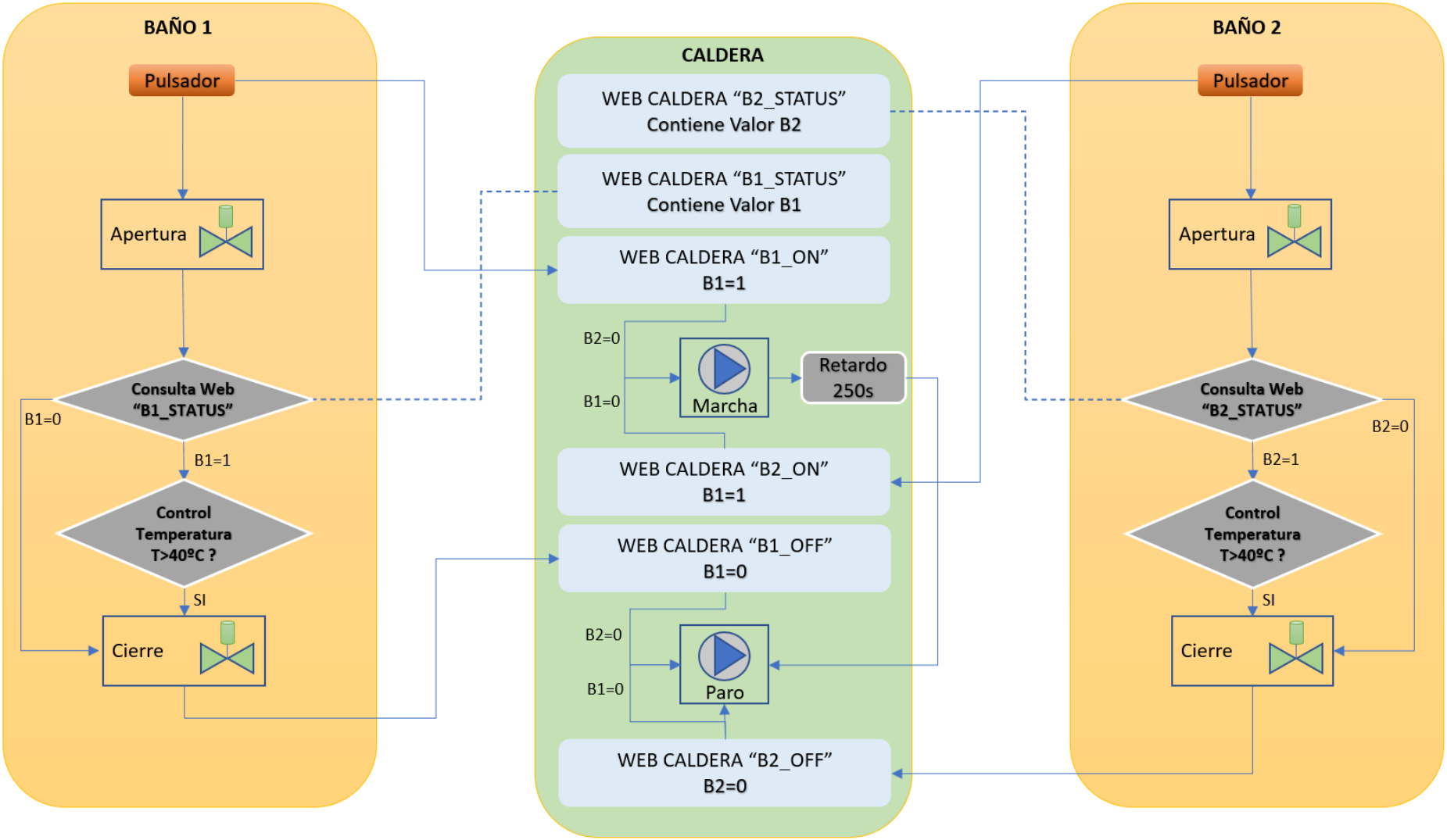

Figura 4.2 Lógica Funcionamiento con Control de Temperatura

Como se puede observar en la figura anterior, tenemos dos cambios principales en la lógica de funcionamiento.

El primero es la eliminación del paso de comprobación del estado del otro baño. Esto es debido a que, en este caso, sí que podemos tener recirculando ambos baños a la vez, ya que, aunque el calentamiento será más lento (menos caudal por cada baño) cada uno cerrará el bypass al llegar a su temperatura de consigna.

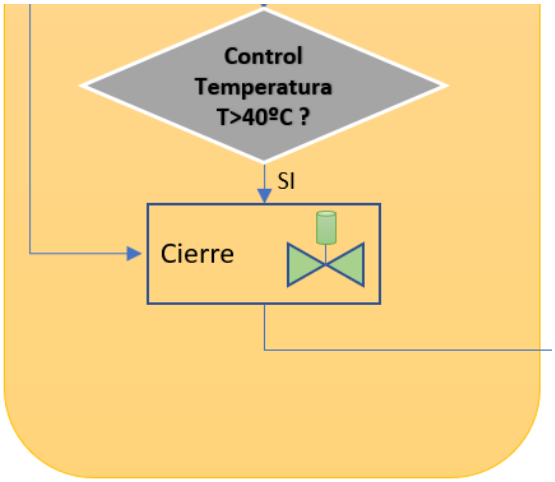

Figura 4.3 Lógica Control de temperatura

Dado que es posible que ambos baños estén recirculando a la vez, el envío de órdenes de arranque y parada de la bomba, se realizarán supervisando el funcionamiento o no del otro baño.

Primero, la RPi de la caldera, antes de enviar una orden de arranque de la bomba, comprueba si el otro baño está recirculando. Si es así, no es necesario enviar orden de arranque de la bomba (ya que esta se encontrará en funcionamiento).

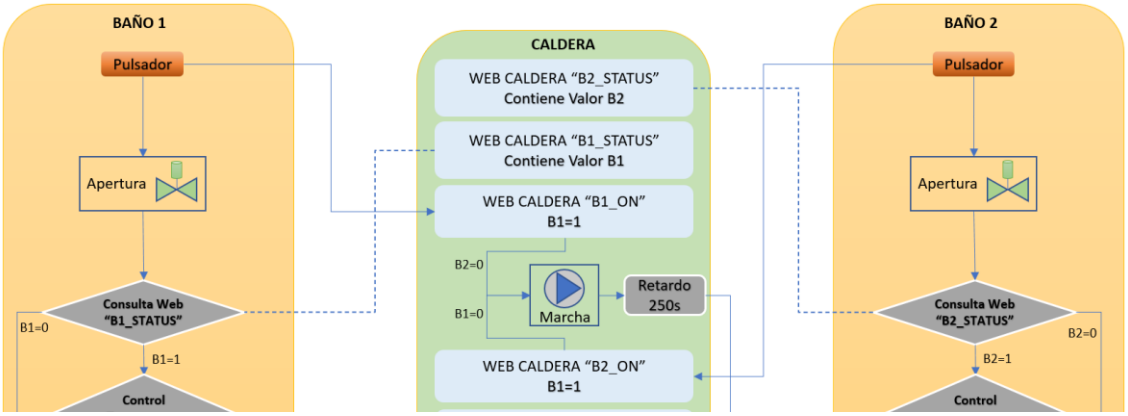

Figura 4.4 Condición Orden de Arranque de Bomba

Y, por otro lado, se incluye la condición de que la orden de parada de la bomba se produzca únicamente cuando el otro baño no esté recirculando.

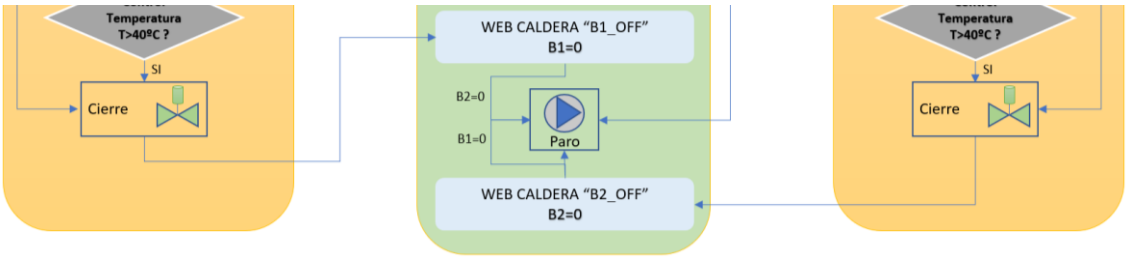

Figura 4.5 Condición Orden de Parada de Bomba

4.3.2. Conexionado de la sonda de temperatura a la Rasbperry Pi

El conexionado de la sonda de temperatura utilizada, como se comentaba en el apartado 3.3.6, se realiza mediante el protocolo de comunicación 1-wire, lo que nos permite un fácil conexionado y obtener una señal digital, sin módulo conversión previa.

Para poder utilizarlo, lo primero que debemos realizar, es habilitar la comunicación "1-wire" de la Raspberry pi en la configuración de la misma.

Para ello, debemos teclear en la consola:

```
pi@raspberrypi:~$ sudo raspi-config
```

En el menú "Interfacing Options", activamos la comunicación "1-wire"

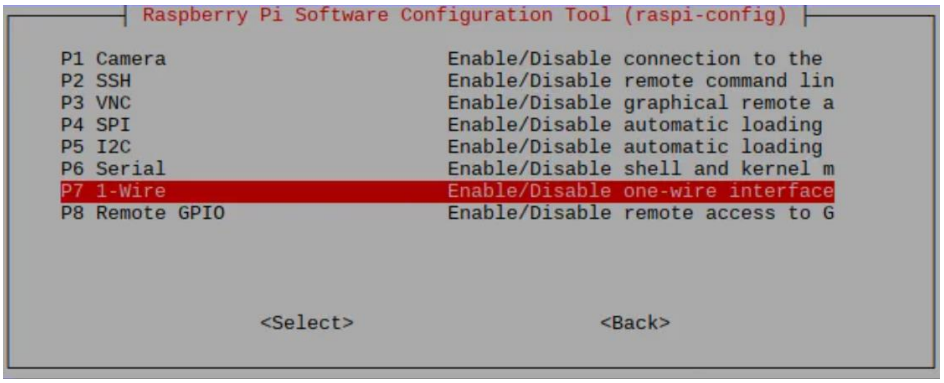

Figura 4.6 Activación "1-wire" en raspi-config

También debemos instalar el paquete de Python 3 "w1thermsensor". Esto lo haremos tecleando en la consola:

```
pi@raspberrypi:~$ sudo pip3 install w1thermsensor
```

Una vez tenemos esto, debemos conectar el sensor de temperatura a la RPi según el siguiente esquema, con una resistencia de 4,7 kΩ (según recomienda el fabricante):

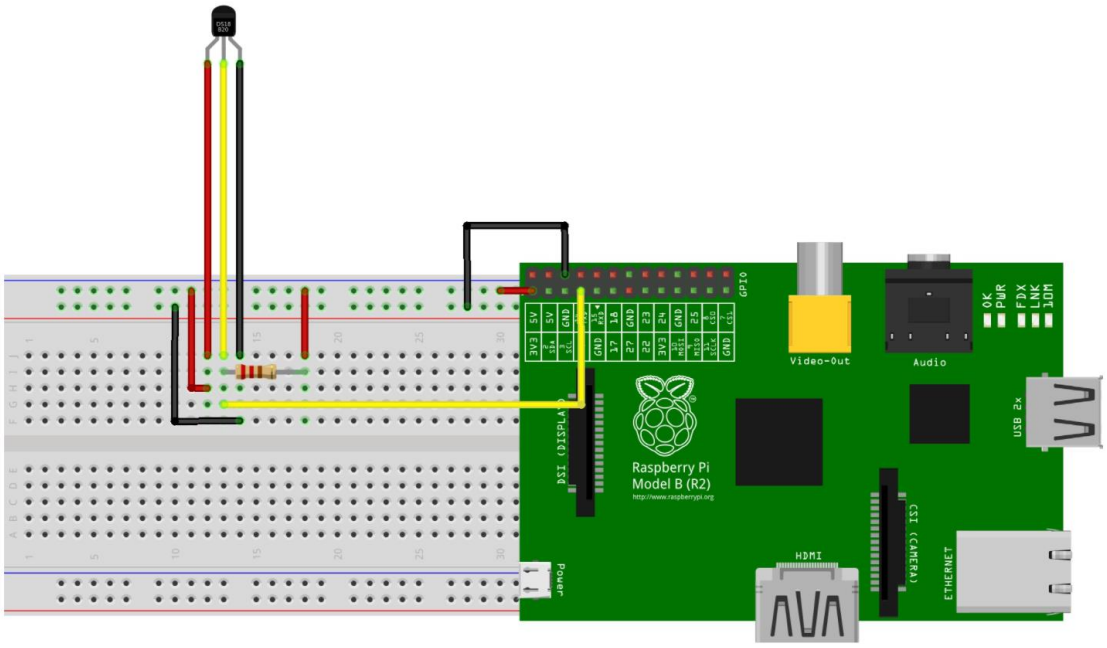

Figura 4.7 Conexión Sensor Temperatura DS18B20 a RPi

Como hemos dicho, este tipo de sensores, tienen la ventaja de poderse conectar varios al mismo GPIO de la placa, diferenciando esta los datos provenientes de cada uno de ellos.

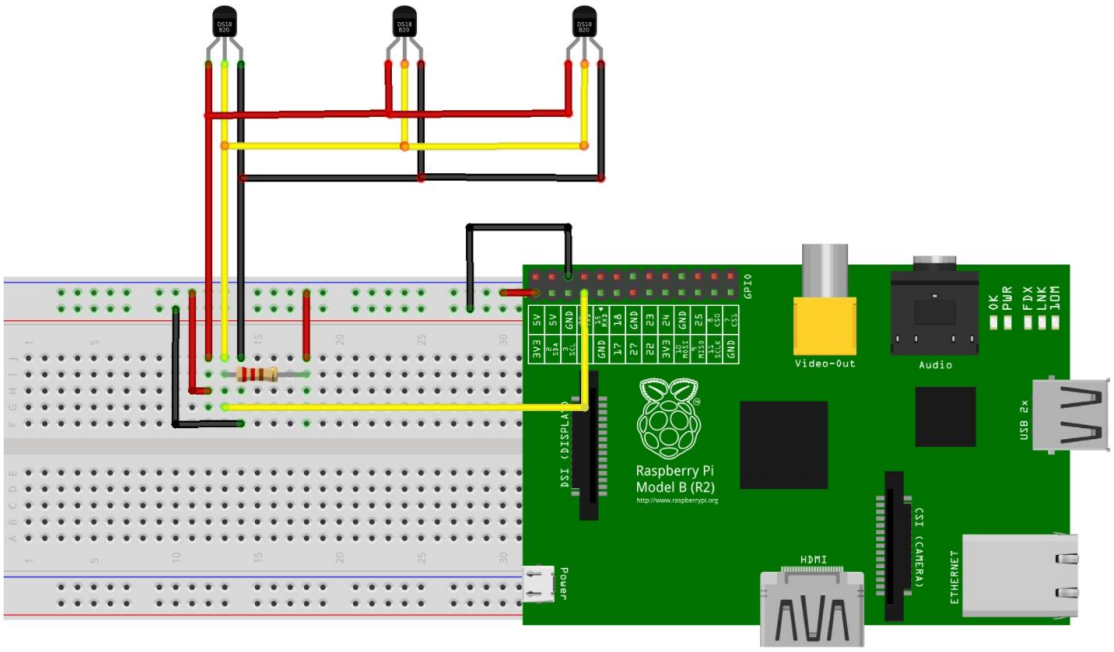

Figura 4.8 Conexión de Varios Sensores de Temperatura DS18B20 a una RPi

Veamos el código necesario para la ejecución de este programa.

4.3.3. Programación Placa Caldera

Primero vamos a analizar los cambios en el código correspondiente a la placa situada en la caldera, haciendo de servidor web y comandando la bomba circuladora.

```
from flask import Flask
import time
import RPi.GPIO as gpio
gpio.setwarnings(False)
gpio.setmode(gpio.BOARD)
gpio.setup(16, gpio.OUT)
gpio.output(16,gpio.LOW)
b1=0
b2=0


app = Flask(__name__)


@app.route('/B1_OFF')
def B1_OFF():
    global b1
    b1=0
    if b2==0:
        gpio.output(16,gpio.LOW)
    return "Bano 1 is OFF"
```

Añadimos esta última parte para parar la bomba únicamente en el caso de que el baño 2 no esté calentando el agua de la tubería.

```
@app.route('/B1_ON')
def B1_ON():
    global b1
    global b2
    b1=1
    if b2==0:
        gpio.output(16,gpio.HIGH)
```

```
    time.sleep(250)
    gpio.output(16,gpio.LOW)
    return "Bano 1 is ON"
```

Tan solo conectamos la bomba sí no ha sido solicitada su activación por el otro baño. También se añade un límite de tiempo en el que la bomba puede estar funcionando, para evitar averías en la misma si se produjera un fallo, no se abriera la válvula de bypass, y estuviera funcionando a caudal cero.

```
@app.route('/B1_STATUS')
def B1_STATUS():
    b1s=str(b1)
    return b1s


@app.route('/B2_OFF')
def B2_OFF():
    global b2
    global b1
    b2=0
    if b1==0:
        gpio.output(16,gpio.LOW)
    return "Bano 2 is OFF"
```

Al igual que en el baño 1, añadimos esta parte para parar la bomba únicamente en el caso de que el baño 1 no esté calentando el agua de la tubería.

```
@app.route('/B2_ON')
def B2_ON():
    global b2
    global b1
    b2=1
    if b1==0:
        gpio.output(16,gpio.HIGH)
    time.sleep(250)
    gpio.output(16,gpio.LOW)
    return "Bano 2 is ON"
```

Tan solo conectamos la bomba sí no ha sido solicitada su activación por el otro baño. También se añade el mismo límite de tiempo en el que la bomba puede estar funcionando.

```
@app.route('/B2_STATUS')
def B2_STATUS():
    b2s=str(b2)
    return b2s

if __name__=="__main__":
    app.run(host='0.0.0.0', port=60, debug=False)
```

4.3.4. Programación Placa Baño 1/2

Y ahora vamos a ver el código de uno de los 2 baños (ya que en ambos casos es idéntico).

```
import time
import urllib.request
import RPi.GPIO as gpio
from w1thermsensor import W1ThermSensor
#Importamos el paquete W1ThermSensor, de la librería instalada anteriormente w1thermsensor para poder acceder a los datos proporcionados por el sensor de temperatura conectado.
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD)

gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)
gpio.setup(16, gpio.OUT)
sensor = W1ThermSensor()
```

Creamos un objeto “sensor”

```
tconsigna=40
```

Establecemos una temperatura de consigna, que será la que pare la recirculación en el baño correspondiente.

```
while True:
```

```
    gpio.wait_for_edge(11,gpio.FALLING)
    temperature = sensor.get_temperature()
```

Esperamos a que se presiones el botón situado en el baño, y una vez que se presiona, obtenemos la temperatura actual del sensor y la almacenamos en la variable “temperature”

```
    if temperature < tconsigna:
        gpio.output(16,gpio.HIGH)
        t=urllib.request.urlopen('http://192.168.0.63:60/B1_ON')
```

Si la temperatura actual, es inferior a la de consigna, ordenamos la apertura de la válvula de bypass y realizamos una petición web a la RPi de la caldera para que active la bomba.

```
estado_b1=urllib.request.urlopen('http://192.168.0.63:60/B1_STATUS')
        b1=int(estado_b1.read())
```

Comprobamos la recepción de la orden de marcha de la bomba, consultando la web de “estado” de la petición.

```
        if b1==1:
            while temperature < tconsigna:
                time.sleep(1)
                temperature=sensor.get_temperature()
                print("the temperature is %s celsius" %temperature)
            u=urllib.request.urlopen('http://192.168.0.63:60/B1_OFF')
            print("Ya se dispone de agua caliente en el BAÑO 1")
            gpio.output(16,gpio.LOW)
```

Si la petición ha sido recibida correctamente (b1=1), se actualiza cada segundo la temperatura percibida por el sensor y se compara con la temperatura de consigna, hasta que la primera es mayor. En ese momento, se envía una petición web de parada de bomba (que solo tendrá efecto en caso de que el otro baño no esté circulando), y se cierra la válvula de bypass.

```
        else:
            gpio.output(16,gpio.LOW)
            u=urllib.request.urlopen('http://192.168.0.63:60/B1_OFF')
```

Si comprobamos que la petición no se recibe correctamente en la placa situada en la caldera, terminamos inmediatamente el proceso cerrando la válvula de bypass y pidiendo la parada de la bomba (aunque no debería estar funcionando debido a la solicitud de este baño, se realiza por seguridad).

4.3.5. Estimación del Coste

A continuación, se resumen los costes principales de este sistema.

Item	Descripción	Ud	P.Unit	Total
3	SISTEMA CON COM. WEB Y SONDAS TEMP	1	243,88 €	243,88 €
3.01	Raspberry Pi Zero WH	3	15,95 €	47,85 €
3.02	Fuente de Alimentación 5V 2A Micro-USB	3	6,10 €	18,30 €
3.03	Tarjeta MicroSDHC 16GB CLASS10	3	5,01 €	15,03 €
3.04	Relé Optoacoplado 10A 230V	3	1,99 €	5,97 €
3.05	Sonda Temperatura DS1280	2	2,37 €	4,74 €
3.06	Bomba Circuladora 10 lpm 5mca	1	109,00 €	109,00 €
3.07	Válvula Solenoide 3/4" 220V	2	11,99 €	23,98 €
3.08	Válvula Antirretorno 1"	1	9,01 €	9,01 €
3.09	Pequeño Material	1	10,00 €	10,00 €

Tabla 4.3 Coste del Sistema Con Comunicación Web y Sondas de Temperatura

En este caso, respecto a la versión anterior, se incrementa el coste por la incorporación de las sondas de temperatura, pero este incremento es de apenas 5 euros, por lo que, viendo las ventajas respecto al anterior, el ratio coste beneficio es muy favorable

5. RESULTADOS

5.1. GENERAL

El resultado obtenido de los sistemas configurados ha sido, en términos generales, satisfactorio.

Aunque las pruebas se han realizado de forma simulada, estas han dado el resultado requerido en cada una de ellas, obviando las limitaciones existentes por el esquema escogido.

El funcionamiento con una única placa nos da la robustez de no necesitar ningún tipo de comunicación con otras placas. Esta simulación no es susceptible de dar problemas si no fallan los elementos que se están utilizando.

El funcionamiento con varias placas y comunicación vía Web asimismo ha funcionado correctamente. Tiene la limitación, al igual que la anterior, de tener la opción de introducir agua caliente en el circuito de fría si se solicita el calentamiento de ambos baños en un intervalo corto de tiempo.

El resultado con tres placas y sensores de temperatura ha resultado el más adecuado de los tres, tanto por la exactitud en el tiempo de recirculación de agua en los baños, como por la robustez ofrecida al poder obtener datos de temperatura de cada uno de los mismos.

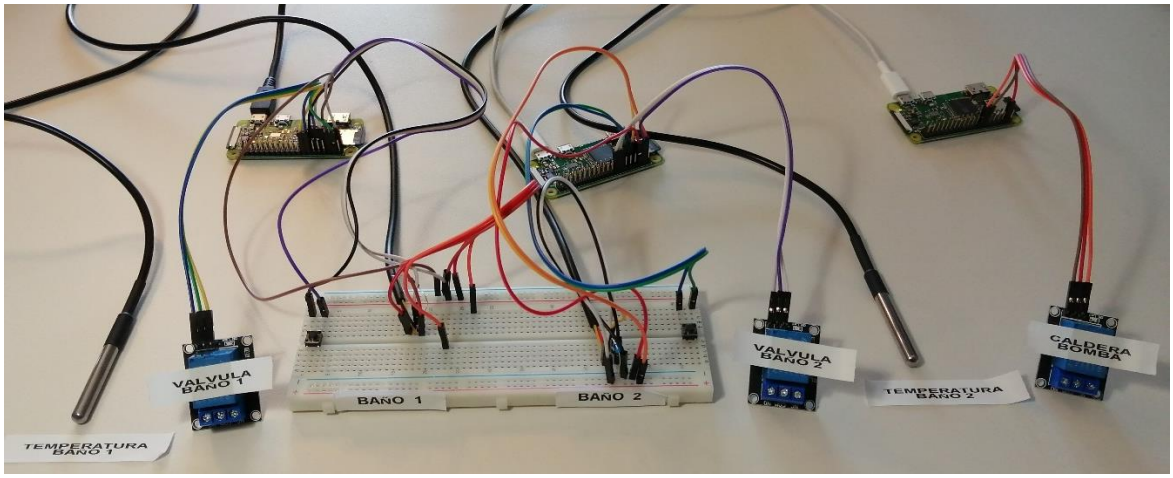

Figura 5.1 Montaje para la simulación con sondas de temperatura

5.2. ECONÓMICO

Si analizamos el resultado desde el punto de vista económico, para el caso considerado en el punto 3.1 del presente trabajo, podemos ver como para un sistema compuesto por dos puntos de consumo, el periodo de amortización es ostensiblemente inferior en este sistema respecto a los modelos comerciales, debido a su bajo coste.

	Kit Básico para un punto de Consumo	Precio por Punto de Consumo Adicional	Coste Para 2 Puntos de Consumo	Ahorro Anual	Periodo de Amortización en años
Aqua Return	347,00 €	347,00 €	694,00 €	37,77 €	18,37
Presto Go	599,50 €	332,30 €	931,80 €	37,77 €	24,67
TFM	198,47 €	45,41 €	243,88 €	37,77 €	6,46

Tabla 5.1 Comparativa Amortización Sistema con 2 puntos de consumo

Igualmente, si comparamos el periodo de amortización para la instalación de otros puntos de consumo adicionales de menor desperdicio de agua, podemos ver como también es muy inferior (90,.82€), pero sigue siendo muy alto como para compensar invertir en ellos.

	Kit Básico para un punto de Consumo	Precio por Punto de Consumo Adicional	Coste Adicional Para 4 Puntos de Consumo	Ahorro Anual	Periodo de Amortización en años
Aqua Return	347,00 €	347,00 €	694,00 €	6,63 €	104,73
Presto Go	599,50 €	332,30 €	664,60 €	6,63 €	100,30
TFM	198,47 €	45,41 €	90,82 €	6,63 €	13,71

Tabla 5.2 Comparativa Amortización Instalación de dos puntos de consumo adicionales para zonas cercanas a la producción de ACS

5.3. MEDIOAMBIENTAL

El resultado obtenido a nivel de consumo de agua, a pesar de no haberse podido probar en una instalación con todos los elementos colocados, resultaría en un ahorro de unos $4m^3$ de agua por persona y año, lo que supone un gran ahorro de agua y una buena contribución a la reducción del consumo a nivel general.

6. CONCLUSIONES Y LÍNEAS FUTURAS DE TRABAJO

6.1. CONCLUSIONES

Una vez concluido el proyecto fin de máster, se puede confirmar que se ha cumplido el objetivo inicial, que consistía en realizar un sistema que consiguiera obtener agua caliente en los baños de una vivienda sin instalación de circulación, sin desperdiciar agua, o al menos reducir significativamente el desecho de la misma.

Los nuevos proyectos del tipo de Raspberry Pi, Arduino, etc., proporcionan muchas posibilidades de automatización de procesos a un coste relativamente bajo.

Hemos podido comprobar cómo es posible realizar una instalación de este tipo, que ya existe en el mercado, con unos costes muy inferiores y por tanto un periodo de amortización muy inferior (del orden de una tercera parte), lo que lo hace un proyecto viable para su instalación en muchas viviendas que no pueden acceder a un sistema comercial de coste más elevado.

Para conseguir realizar el presente trabajo, ha sido necesaria la adquisición de numerosos conocimientos en el ámbito de la programación en Python, además del sistema Raspberry Pi que, a pesar de no haber estado aplicados a la robótica, proporcionan una base importante para futuros desarrollos en este ámbito.

Otra conclusión que podemos sacar del trabajo es la posibilidad de mejorar la robustez del sistema. Teniendo en cuenta que la configuración óptima de funcionamiento para este sistema requiere comunicación vía Wifi entre las diferentes placas controladoras, debemos tener en cuenta las posibilidades de error que se pueden producir.

6.2. LÍNEAS FUTURAS DE TRABAJO

A continuación, se indican las futuras líneas de trabajo que, debido al alcance del mismo, no se han llevado a cabo.

En primer lugar, como se comenta en el punto anterior, se estudiarán métodos para hacer más robusta la programación del sistema, haciendo hincapié en dos puntos principales:

En primer lugar, se ha de contemplar los posibles fallos de comunicación de los equipos. En este trabajo se ha tenido en cuenta algún posible fallo de comunicación, pero no se ha estudiado la solución de los mismos, de cara a la posible implementación de este sistema en un producto comercial.

Por otra parte, al hilo de lo anterior, se debería incorporar un sistema de comunicación con el usuario mediante medios visuales y sonido. Se podrían incorporar leds de estado en el equipo que indicaran:

- Encendido

- Conectividad Wifi
- Disponibilidad de agua caliente
- Actividad de la recirculación

Dado que el equipo se encontraría bajo un lavabo u otro sanitario y que la visibilidad es baja, se podría incorporar un altavoz que, bien mediante un código de tonos o bien mediante una locución, cualquiera de los estados, o fallos del sistema.

Otra de las vías de ampliación del trabajo, sería la activación de los sistemas mediante interruptores domóticos, tipo zigbee. De esta forma, sería posible la activación del sistema mediante un sistema domótico integrado, que nos permitiría realizar todas las operaciones incluso mediante comandos de voz con Alexa, Siri, Google Nest, etc.

Otra de las posibles líneas de ampliación, sería el estudio de implementación de este sistema en un equipo comercial. Para ello se requeriría un estudio de cómo incorporar los elementos de cada placa controladora y elementos mecánicos en un único contenedor adaptado.

Por último, de cara a una configuración como un producto comercial, se podría intentar configurar una de las Raspberry Pi como punto de acceso Wifi, de forma que el producto fuera “Plug and Play”, funcionando con una red Wifi independiente a la del lugar en el que se instale.

7. BIBLIOGRAFÍA

Aigües d'Elx. (s.f.). *Informe de desarrollo sostenible 2019.* Obtenido de https://www.aigueselx.com/informe-de-desarrollo-sostenible

aprendiendoarduino.wordpress.com. (s.f.). Obtenido de https://aprendiendoarduino.wordpress.com/2020/03/04/manejar-gpio-raspberry-pi/

Aqua Return. (s.f.). *www.aquareturn.com.* Obtenido de www.aquareturn.com: www.aquareturn.com

Díez, G. (s.f.). *Construible*. Obtenido de https://www.construible.es/

Grundfos. (s.f.). Obtenido de https://product-selection.grundfos.com/

https://www.raspberrypi.org/software/. (s.f.). Obtenido de https://www.raspberrypi.org/software/

IONOS. (s.f.). *ionos.es.* Obtenido de https://www.ionos.es/digitalguide/servidores/know-how/servidor-lamp-la-solucion-para-webs-dinamic

Ministerio de fomento. (s.f.). *CTE.* Obtenido de CTE: codigotecnico.org

Prestoiberica. (s.f.). *Prestoiberica Presto Go System.* Obtenido de https://www.prestoiberica.com/presto-go-system/

Tarifa Presto. (s.f.). *Tarifa Presto.* Obtenido de Tarifa Presto: https://www.prestoiberica.com/wp-content/themes/prestoiberica/assets/docs/catalogs/Catalogo-PrestoIberica-2020-tarifa.pdf

Wikipedia. (s.f.). Obtenido de https://es.wikipedia.org/wiki/Raspberry_Pi#Raspberry_Pi_Zero:_modelos

8. ANEXOS

8.1. CÓDIGO

8.1.1. Implementación mediante una placa y cableado

```
import time
import RPi.GPIO as gpio
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD)
gpio.setup(3, gpio.OUT)
gpio.setup(5, gpio.OUT)
gpio.setup(7, gpio.OUT)
gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)
gpio.setup(16, gpio.IN,pull_up_down=gpio.PUD_UP)

while True:
    boton1=gpio.input(11)
    boton2=gpio.input(16)



    if boton1==1 and boton2==1:
        gpio.output(3,gpio.LOW)
        gpio.output(5,gpio.LOW)
        gpio.output(7,gpio.LOW)

    else:
        if boton1==0:
            gpio.output(3,gpio.HIGH)
            gpio.output(5,gpio.HIGH)
            time.sleep(150)
```

```
        gpio.output(3,gpio.LOW)
        gpio.output(5,gpio.LOW)
    else
        gpio.output(3,gpio.HIGH)
        gpio.output(7,gpio.HIGH)
        time.sleep(80)
        gpio.output(3,gpio.LOW)
        gpio.output(7,gpio.LOW)
```

8.1.2. Implementación mediante 3 placas y comunicación Web

8.1.2.1. Rasberry Pi Caldera

```
from flask import Flask
import time
import RPi.GPIO as gpio
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD) # En este caso Utilizamos la numeración física

gpio.setup(16, gpio.OUT) #Este pin lo utilizamos para activar el relé de la bomba.
gpio.output(16,gpio.LOW)
b1=0
b2=0

app = Flask(__name__)

@app.route('/B1_OFF')
def B1_OFF():
    gpio.output(16,gpio.LOW)
    global b1
```

```
    b1=0
    return "Bano 1 is OFF"

@app.route('/B1_ON')
def B1_ON():
    global b1
    b1=1
    gpio.output(16,gpio.HIGH)
    time.sleep(150)
    gpio.output(16,gpio.LOW)
    return "Bano 1 is ON"

@app.route('/B1_STATUS')
def B1_STATUS():
    b1s=str(b1)
    print (b1s)
    return b1s

@app.route('/B2_OFF')
def B2_OFF():
    gpio.output(16,gpio.LOW)
    global b2
    b2=0
    return "Bano 2 is OFF"

@app.route('/B2_ON')
def B2_ON():
    global b2
    b2=1
    gpio.output(16,gpio.HIGH)
    time.sleep(150)
    gpio.output(16,gpio.LOW)
```

```
    return "Bano 2 is ON"

@app.route('/B2_STATUS')
def B2_STATUS():
    b2s=str(b2)
    return b2s

if __name__=="__main__":
    app.run(host='0.0.0.0', port=60, debug=False)
```

8.1.2.2. Rasberry Pi Baño 1

```
import time
import urllib.request
import RPi.GPIO as gpio
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD) # En este caso Utilizamos la numeración física

gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP

gpio.setup(16, gpio.OUT)

gpio.output(16,gpio.LOW)

while True:

    gpio.wait_for_edge(11,gpio.FALLING)
    estado_b2=urllib.request.urlopen('http://192.168.0.63:60/B2_STATUS')
    b2=int(estado_b2.read())
    if b2!=1:
        gpio.output(16,gpio.HIGH)
        t=urllib.request.urlopen('http://192.168.0.63:60/B1_ON')
```

```
        time.sleep(1)
        estado_b1=urllib.request.urlopen('http://192.168.0.63:60/B1_STATUS')
        b1=int(estado_b1.read())
        if b1==1:
            time.sleep(20)
            t=urllib.request.urlopen('http://192.168.0.63:60/B1_OFF')
            gpio.output(16,gpio.LOW)
        else:
            gpio.output(16,gpio.LOW)
            t=urllib.request.urlopen('http://192.168.0.63:60/B1_OFF')
```

8.1.2.3. Rasberry Pi Baño 2

```
import time
import urllib.request
import RPi.GPIO as gpio
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD) # En este caso Utilizamos la numeración física

gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)
gpio.setup(16, gpio.OUT)

gpio.output(16,gpio.LOW)

while True:

    gpio.wait_for_edge(11,gpio.FALLING)
    estado_b1=urllib.request.urlopen('http://192.168.0.63:60/B1_STATUS')
    b1=int(estado_b1.read())
    if b1!=1:
        gpio.output(16,gpio.HIGH)
        t=urllib.request.urlopen('http://192.168.0.63:60/B2_ON')
```

```
        time.sleep(1)
        estado_b2=urllib.request.urlopen('http://192.168.0.63:60/B2_STATUS')
        b2=int(estado_b2.read())
        if b2==1:
            time.sleep(20)
            t=urllib.request.urlopen('http://192.168.0.63:60/B2_OFF')
            gpio.output(16,gpio.LOW)
        else:
            gpio.output(16,gpio.LOW)
            t=urllib.request.urlopen('http://192.168.0.63:60/B2_OFF')
```

8.1.3. Utilización de Sensores de Temperatura

8.1.3.1. Rasberry Pi Caldera

```
from flask import Flask
import time
import RPi.GPIO as gpio
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD)
gpio.setup(16, gpio.OUT)
gpio.output(16,gpio.LOW)

b1=0
b2=0

app = Flask(__name__)

@app.route('/B1_OFF')
def B1_OFF():
    global b1
```

```
    global b2
    b1=0
    b2s=str(b2)
    print (b2s)
    if b2==0:
        gpio.output(16,gpio.LOW)
    return "Bano 1 is OFF"

@app.route('/B1_ON')
def B1_ON():
    global b1
    global b2
    b1=1
    if b2==0:
        gpio.output(16,gpio.HIGH)
    time.sleep(250)
    gpio.output(16,gpio.LOW)
    return "Bano 1 is ON"

@app.route('/B1_STATUS')
def B1_STATUS():
    b1s=str(b1)
    return b1s

@app.route('/B2_OFF')
def B2_OFF():
    global b2
    global b1
    b2=0
    b1s=str(b1)
    print (b1s)
    if b1==0:
```

```
        gpio.output(16,gpio.LOW)
    return "Bano 2 is OFF"
@app.route('/B2_ON')
def B2_ON():
    global b2
    global b1
    b2=1
    if b1==0:
        gpio.output(16,gpio.HIGH)
    time.sleep(250)
    gpio.output(16,gpio.LOW)
    return "Bano 2 is ON"

@app.route('/B2_STATUS')
def B2_STATUS():
    b2s=str(b2)
    return b2s

if __name__=="__main__":
    app.run(host='0.0.0.0', port=60, debug=False)
```

8.1.3.2. Rasberry Pi Baño 1

```
import time
import urllib.request
import RPi.GPIO as gpio
from w1thermsensor import W1ThermSensor
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD)
gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)
gpio.setup(16, gpio.OUT)
```

```
sensor = W1ThermSensor()

tconsigna=40



while True:

    gpio.wait_for_edge(11,gpio.FALLING)
    temperature = sensor.get_temperature()

    if temperature < tconsigna:
        gpio.output(16,gpio.HIGH)
        t=urllib.request.urlopen('http://192.168.0.63:60/B1_ON')
        estado_b1=urllib.request.urlopen('http://192.168.0.63:60/B1_STATUS')
        b1=int(estado_b1.read())
        if b1==1:
            while temperature < tconsigna:
                time.sleep(1)
                temperature=sensor.get_temperature()
                print("the temperature is %s celsius" %temperature)
            u=urllib.request.urlopen('http://192.168.0.63:60/B1_OFF')
            print("Ya se dispone de agua caliente en el BAÑO 1")
            gpio.output(16,gpio.LOW)
        else:
            gpio.output(16,gpio.LOW)
            u=urllib.request.urlopen('http://192.168.0.63:60/B1_OFF')
```

8.1.3.3. Rasberry Pi Baño 2

```
import time
import urllib.request
```

```
import RPi.GPIO as gpio
from w1thermsensor import W1ThermSensor
gpio.setwarnings(False)

gpio.setmode(gpio.BOARD)

gpio.setup(11, gpio.IN,pull_up_down=gpio.PUD_UP)
gpio.setup(16, gpio.OUT)
sensor = W1ThermSensor()
tconsigna=40

while True:

    gpio.wait_for_edge(11,gpio.FALLING)
    temperature = sensor.get_temperature()

    if temperature < tconsigna:
        gpio.output(16,gpio.HIGH)
        t=urllib.request.urlopen('http://192.168.0.63:60/B2_ON')
        estado_b2=urllib.request.urlopen('http://192.168.0.63:60/B2_STATUS')
        b2=int(estado_b2.read())
        if b2==1:
            while temperature < tconsigna:
                time.sleep(1)
                temperature=sensor.get_temperature()
                print("the temperature is %s celsius" %temperature)
            u=urllib.request.urlopen('http://192.168.0.63:60/B2_OFF')
            print("Ya se dispone de agua caliente en el BAñO 2")
            gpio.output(16,gpio.LOW)
        else:
            gpio.output(16,gpio.LOW)
            u=urllib.request.urlopen('http://192.168.0.63:60/B2_OFF')
```