Ingeniería de Sistemas Electrónicos

Casa Domótica

Curso 2022-2023

Bloque 2

Autores:

Hao Feng Chen Fu

Álvaro Rodriguez Piñeiro

Jaime Ruiz López

Gonzalo Santa Cruz del Río

Valeriu Petre Stanca

Índice del documento

1 OBJETIVOS DE LA PRÁCTICA

1.1 Especificaciones iniciales del diseño a construir

El proyecto de la casa domótica debe incluir obligatoriamente los siguiente bloques funcionales:

- Gestión del servidor Web con una interfaz amigable
- Sincronización por NTP y mantener la hora del sistema con RTC
- Sistema autónomo alimentado por baterías
- Incluir un modo de bajo consumo
- Incluir algún subsistema analógico de mediana complejidad montado sobre una PCB
- Medida del consumo de la aplicación
- Almacenamiento de información en la memoria flash

En adición a lo anterior, el proyecto de la casa domotizada incluye los siguientes módulos para simular una casa:

- Sensor de luminosidad para la intensidad de la luces
- Sensor de temperatura y humedad
- Sensor de presencia para encender las luces
- Sensor sísmico
- Mando a distancia
- Garaje
- Termostato, que en este caso es un ventilador para regular la temperatura

1.2 Especificaciones finales del sistema diseñado y construido

Durante el desarrollo del proyecto se ha completado las especificaciones inicales, pero se fueron implementando las siguientes variaciones:

Módulo de alimentación:

- Disipadores, ya que los transistores alcanzaban altas temperaturas.

Módulo del garaje:

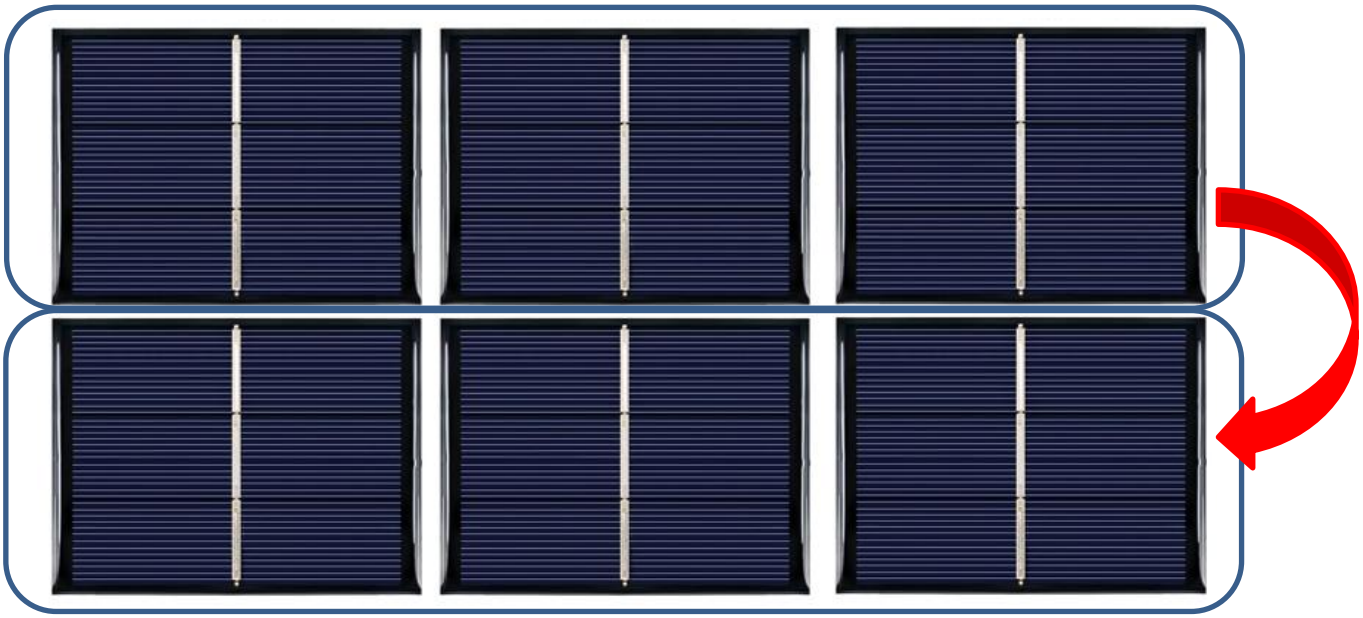
- Correción de soldadura de componentes en la PCB del mando, debido a un problema en la asignación inicial de los componentes en la misma.
- Se ha incluido un Time Out para la puerta del garaje que automatiza el proceso una vez pasan 5 segundos de la última activación de la misma, y estando la puerta subida.

Módulo del detector sismico:

- Correción de soldadura de componentes en la PCB del sensor, debido a un problema en la asignación inicial de los componentes en la misma.
- Se incluye un interruptor que permita inciar la vibración del sistema mediante el LM555.

Células Solares:

- Se rediseña el sistema de alimentación solar, con 6 nuevos paneles, estructurados en serie y en paralelo para obtener un mejor rendimiento ya que el anterior diseño era insuficiente para alimentar las baterías de una mantera eficiente.



El color azul indica las placas que se han asociado en serie y la flecha roja, la asociación en paralelo de los dos bloques asociados en serie, entregando así una tension a las baterías de 10.25v en vacio.

Baterías:

- Se incluye un interruptor que permita inciar el funcionamiento de las pilas.

1.3 **Acrónimos utilizados**

PIR	Passive InfraRed
GPIO	General Purpose Input/Output
I2C	Inter-Integrated Circuit
SPI	Serial Peripherical Interface
ISR	Interrupt Service Routine
PWM	Pulse Width Modulation
LCD	Liquid Cristal Display
LED	Light-Emitting Diode
ADC	Analog-to-Digital Converter
SW	Software
HW	Hardware

1.4 Tiempo empleado en la realización de la práctica.

[Tiempo empleado para realizar la práctica]: El tiempo total empleado ha sido de 384 horas aproximadamente.

1.5 Bibliografía utilizada

[RD1] Manual de usuario, Carmine Noviello – Mastering STM32

[RD2] Datasheet, SHT30; https://www.alldatasheet.es/datasheet-pdf/pdf/897974/ETC2/SHT30.html

[RD3] Datasheet, HC-SR501

[RD4] Datasheet, HXYP-2S-A18 2S

[RD5] Datasheet,LM555;https://pdf1.alldatasheet.com/datasheetpdf/view/53587/FAIRCHILD/LM555.html

[RD6] Datasheet,LM741;https://pdf1.alldatasheet.com/datasheetpdf/view/53589/FAIRCHILD/LM741.html

[RD7] Datasheet,ACS712;https://pdf1.alldatasheet.com/datasheetpdf/view/168326/ALLEGRO/ACS712.html

[RD8]

1.6 Autoevaluación.

- ✓ Identificar en un documento de especificaciones técnicas de un sistema electrónico los requisitos técnicos necesarios para plantear diferentes alternativas tecnológicas para la implementación práctica del mismo.
- ✓ Desarrollar un sistema electrónico de mediana complejidad combinando diferentes tecnologías
- ✓ Construir sistemas electrónicos utilizando PCBs aplicando las técnicas de diseño adecuadas a la tipología del diseño
- ✓ Emplear la instrumentación de laboratorio, las herramientas de desarrollo y depuración comerciales para la integración y puesta a punto de circuitos y sistemas electrónicos.
- ✓ Saber generar documentación precisa y transferible sobre el sistema desarrollado.

Dificultades encontradas durante el diseño del proyecto:

- Una de las dificultades que hemos encontrado a la hora de realizar el proyecto es el tema de la alimentacion, ya que debiamos encontrar una manera de alimentar todos los subsistemas del proyecto haciendo uso de alimentacion externa. Para este proposito hemos elegido un par de baterias 18650 puestas en serie, ya que con esto obteniamos 7.4V de alimentacion (8.4V en vacio). Junto con esto hemos elgido una celula fotovoltaica para cargar dichas baterias, junto a un modulo de carga (HXYP-2S-A18 2S) para evitar tanto sobrecargas como sobredescargas. El problema real que encontramos es el de conseguir la tensión suficiente como para cargar dichas

baterias, ya que la celula nos entregaba 5V y el modulo de carga demandaba 9V para cargar las baterias. Finalmente decidimos comprar un par de celulas mas y conectarlas en serie junto a la otra para conseguir la tensión minima necesaria.

- Otra dificultad encontrada durante el desarrollo del proyecto ha sido a la hora del diseño y soldado de las placas de circuito impreso, ya que durante el diseño de estas se cometieron un par de errores como son el olvido a la hora de colocar un condensador en una placa y una pista necesaria en otra de las placas. Ambos errores fueron facilmente subsanables y no hubo mayor problema.
- La mayor dificultad encontrada, por lo menos en la parte HW, fue el tema del acondicionamiento del sensor piezoelectrico, ya que era un componente poco trabajado y del que poseaimos pocos conocimientos. Tras unas intensas busquedas y despues de informarnos acerca del funcionamiento de estos, conseguimos acondicionarlo de manera adecuada y ponerlo a trabajar de la forma que queriamos.
- El trabajo con el piezoeléctrico ha sido un proceso tedioso. Se ha utilizado una superficie metálica para la recepción de las vibraciones, tanto de un piezo eléctrico directamente, como de la recepción de otro piezoeléctrico debido a la vibración de uno de estos emitendo a tráves del aestable multivibrador LM555 a 3KHz. La tensión de salida de los piezoeléctricos ha sufrido variaciones durante todo el proceso, siendo al principio del desarrollo del proyecto, alrededor de 1V en el caso del piezo que recibia vibraciones a traves de golpes en la propia superficie, y de unos 800 mV el que recibía, pasando por el operacional no inversor de ganancia 2. En las últimas semanas del proyecto encontramos que fuimos capaces de sacar incluso tensiones de hasta 10 V con pequeños impatos en la superficie metálica, lo cual fue algo que nos sorprendió. Otra de las conclusiones que fuimos obteniendo fue, que hay una cierta distancia en la que si hay dos piezoeléctricos muy cerca, la recepción de vibración es muy pequeña, y no es posible tener a la salida del operacional una señal funcional, lo suficientemente grande para poder analizarla con el microprocesador.
- El manejo del servidor web ha sido complejo porque no hay prácticamente información sobre los ficheros cgi y cgx, esto ha dificultado la implementación final, tanto en la representación como en la interacción de la página web.

2 RECURSOS UTILIZADOS DEL MICROCONTROLADOR

2.1 Diagrama de bloques hardware del sistema.

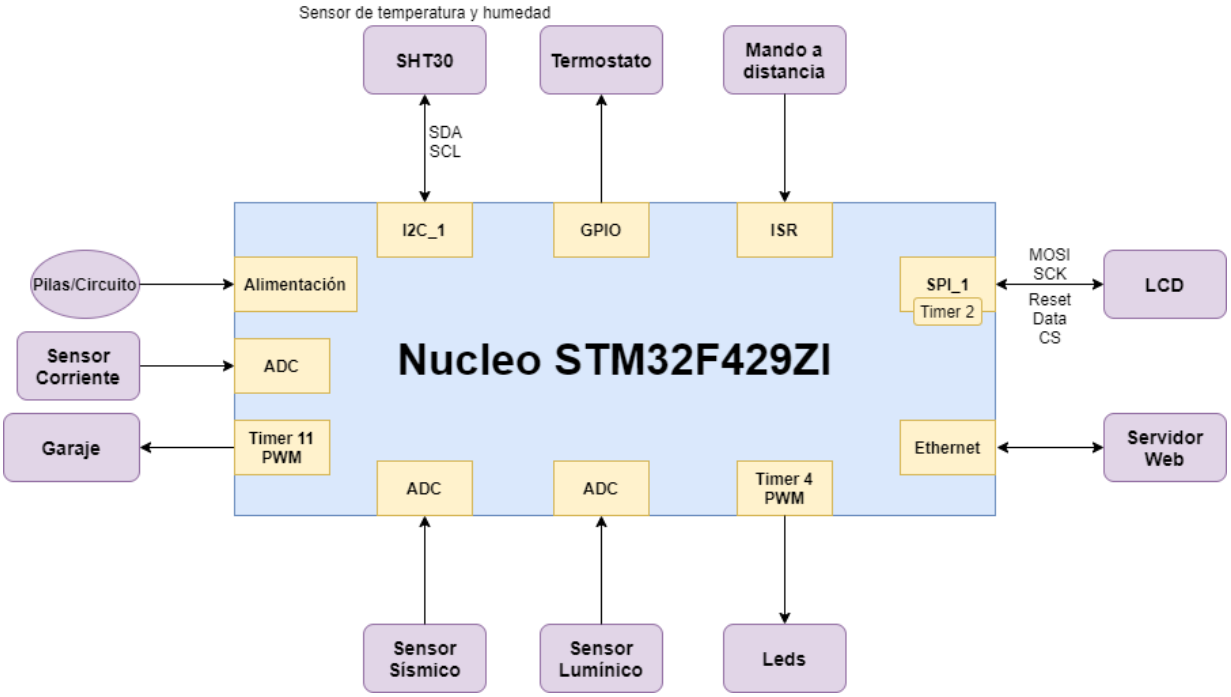


Figura 1: Conexión hardware

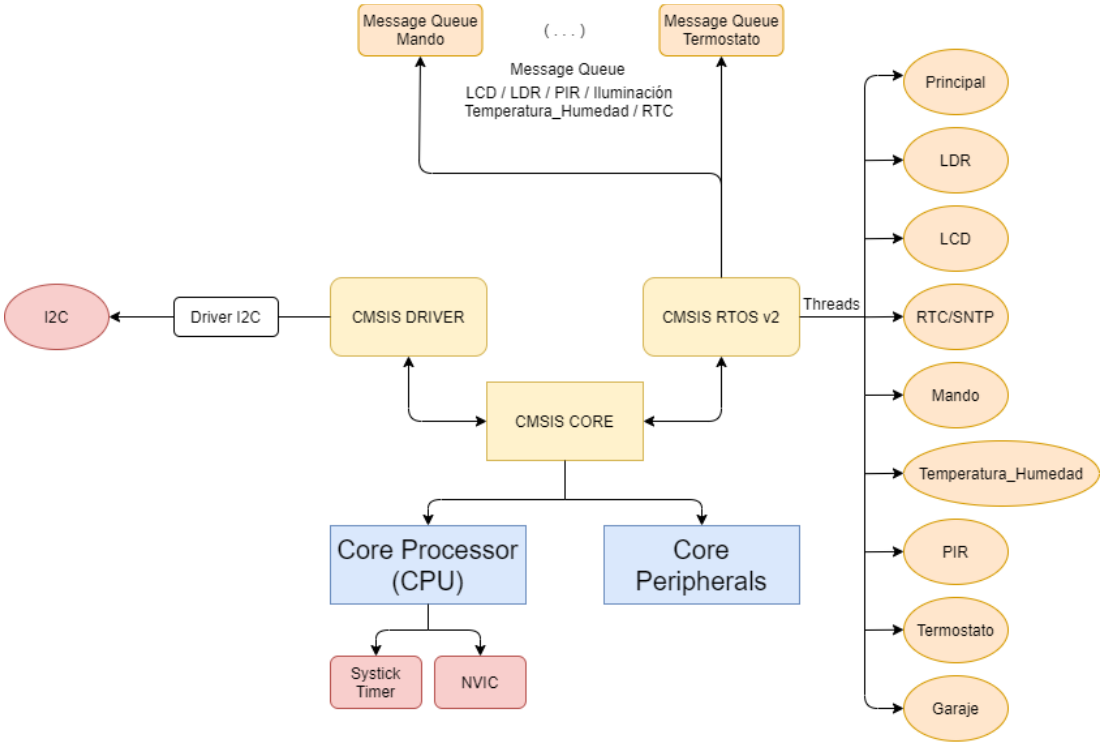


Figura 2: Conexiones del CMSIS

Pines STM32F429ZI	Funcionalidad
PE11	Interrupción PIR
PD12	Interrupción mando a distancia
PC13	Interrupción del botón de usuario
PC3	ADC del LDR
PC0	ADC del sensor de corriente
PA3	ADC del piezoeléctrico
PD15	Salida PWM para la iluminación
PB8	Pin SCL para el sensor de temperatura y humedad
PB9	Pin SDA para el sensor de temperatura y humedad
PC6	Salida para encender el ventilador
PF7	Servomotor de las puertas del garaje

Tabla 1: Conexión de los pines

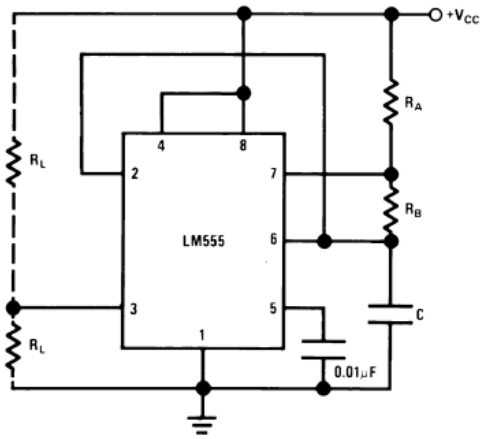
Pines STM32F429ZI	Pines mbed	Funcionalidad
+ 3,3 V	**DIP40**	Alimentación
GND	**DIP1**	Masa
PB5	**DIP5**	MOSI
PA5	**DIP7**	SCK
PD14	**DIP11**	CS
PF13	**DIP8**	A0
PA6	**DIP6**	RESET

Tabla 2: Conexión entre la tarjeta NUCLEO STM32F429Zi y la mbed app board

3 DESARROLLO DE SUBSISTEMAS

3.1 **Analógico**

<u>Módulo del mando a distancia:</u>

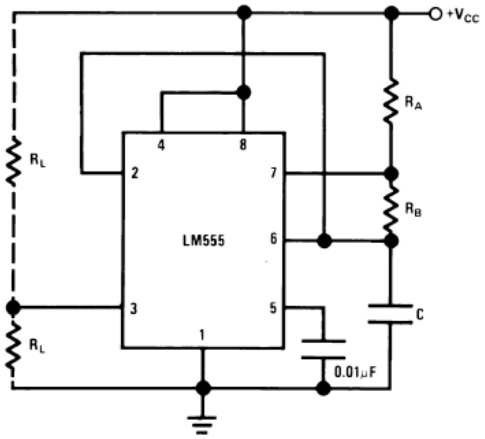


$$f = \frac{\ln(2)}{(R_A + 2R_B) * C} =$$

$$= \frac{1.44}{(4.99k + (2 * 3.3k)) * 3.3n} = 37.65\ KHz$$

En este modo de funcionamiento, aestable, el circuito funciona como un multivibrador. El condensador externo se carga a través de RA + RB y se descarga a través de RB, para poder lograr establecer un ciclo de trabajo preciso. Esto es justo lo que buscamos con esta configuración ya que partiendo de una señal continua, en este caso proporcionada por dos pilas de botón puestas en serie para garantizar el correcto funcionamiento del circuito (6 V), obtendríamos en la salida del mismo una señal periódica con una frecuencia de 37,65 KHz, estando este, en el márgen de recepción del led infrarrojo, que recibirá esta señal periódica y habilitará el funcionamiento de la puerta del garaje.

Módulo del Sensor Sísmico:



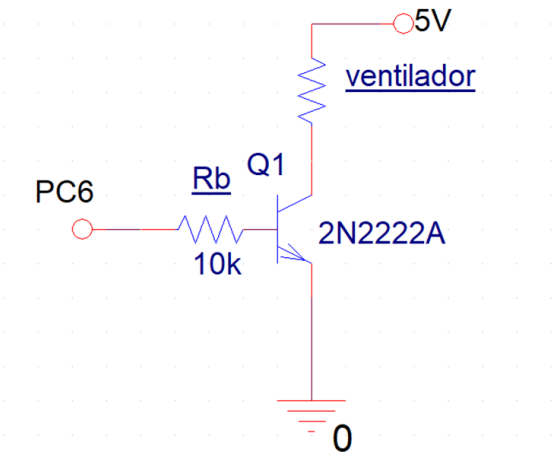
$$f = \frac{\ln(2)}{(R_A + 2R_B) * C} =$$

$$= \frac{1.44}{(9.2k + (2 * 240)) * 47n} = 3.17\ KHz$$

En este modo de funcionamiento, aestable, el circuito funciona como un multivibrador. El condensador externo se carga a través de RA + RB y se descarga a través de RB, para poder lograr establecer un ciclo de trabajo preciso. Esto es justo lo que buscamos con esta configuración ya que partiendo de una señal continua, en este caso proporcionada por una alimentación que obtendrémos de una de las lineas de el módulo de alimentación, obtendríamos en la salida del mismo una señal periódica con una frecuencia de 3,17 KHz, estando este, osclinado en la frecuencia fundamental del piezoeléctrico.

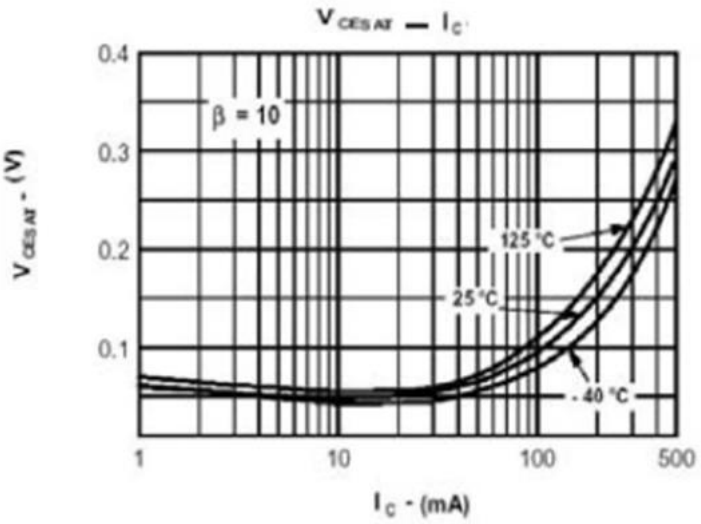
Módulo del termostato:

El objetivo consiste en "regular" la temperatura de la casa, la cual esta recogida por un sensor de temperatura y humedad que se comunica con nuestro microprocesador por I2C. La temperatura, se consigue "regular" mediante un ventilador que funciona con 5V a 20 mA. El esquema electrónico es el siguiente:



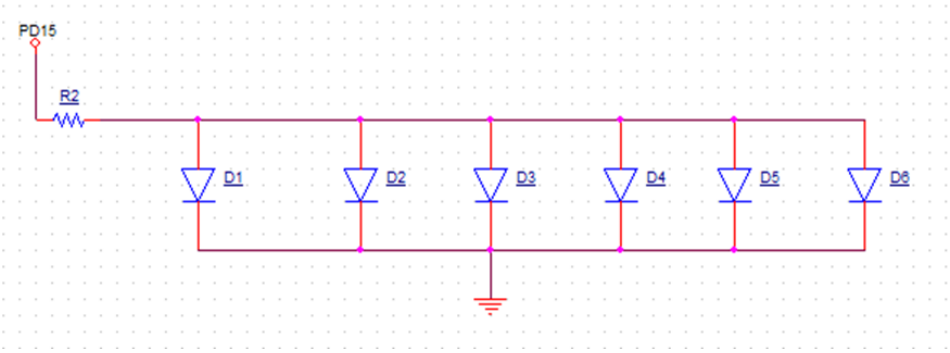
Como nuestro ventilador funciona a 5V, no podemos conectar directamente el ventilador a un pin digital de la placa ya que como máximo tendríamos una tensión de 3,3V. Esto no será suficiente para excitar a nuestro

ventilador. Es por ello, que utilizamos este transistor en conmutación para excitar nuestro ventilador. Además, hemos incluido una resistencia en la base para limitar la corriente por el transistor. De esta manera con **Rb=10KΩ**, conseguimos limitar la corriente del colector a **56 mA.** Con esta corriente, sabemos que la tensión que cae en el transistor Vce es de **50 mV** como se puede ver en el datasheet:



De tal manera que estamos saturando el transistor y permitiendo una mejora en el funcionamiento del ventilador. Ya que ahora si funciona con la tensión de funcionamiento del ventilador (5 V y una corriente por encima de los 20 mA). En base a esto, cuando pongamos un “1” en la base del transistor, el ventilador se activará y cuando pongamos un “0” el ventilador se apagará.

Módulo de los LEDs de la casa:

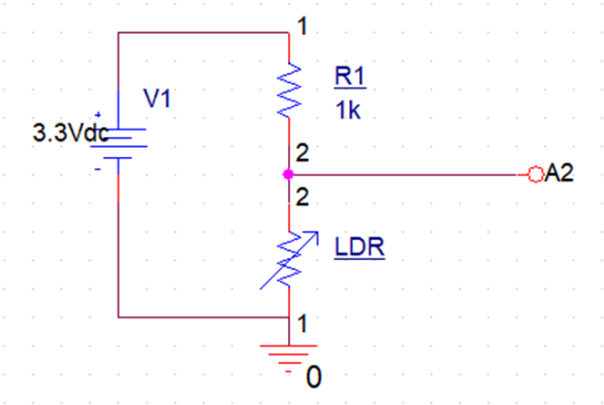


Esta formado por seis LEDs en paralelo y una resistencia para limitar la corriente que circula por el circuito. Este circuito se excitará con el **PD15**, que esta configurado para que funcione con una función alternativa (**AF2_TIM4**). Esto permite emitir por el pin una señal PWM, la cual excitará los LEDs. Por otra parte, la resistencia R2, se utiliza para limitar la corriente que circula a través de los LEDs y por tanto ahorrar en el consumo de la batería. Para hallar el valor de esta resistencia, pusimos un ciclo de trabajo del 100% en el pin PD15. De esta manera obtendremos la corriente máxima que extraer del pin. Sabiendo que la tensión del pin es de **3,3 V** y la tensión de conducción por el diodo es de **2,62 V**. Establecemos una corriente máxima de **14 mA**, con esta corriente los leds se encienden

de forma correcta. De tal forma, que necesitamos una **R2=50Ω.** En función del modulo del LDR y del PIR gestionaremos el encendido de las luces de la casa.

Módulo del LDR:

El LDR está acondicionado con una resistencia en serie con el objetivo de limitar la corriente. De tal manera que el circuito utilizado para la gestión de las medidas de luminosidad es el siguiente:



De esta manera, a partir de un pin analógico (A2), el microprocesador procesa la tensión recibida en el LDR. La cual cambiará en función del grado de oscuridad en el ambiente.Por otra parte, el consumo de corriente máxima de este circuito es de:

$$Imax = \frac{3{,}3}{LDRmin + R1} = \frac{3.3}{71 + 1000} = \mathbf{3.08\ mA}$$

En función de la tensión que cae en el LDR, gestionamos el ciclo de trabajo del PWM que permitirá encender los LEDs. El ciclo de trabajo se gestiona con cuatro posibles rangos de valores en función de la luminosidad del ambiente.

3.2 **Alimentación**

La alimentacion integra del sistema se realiza mediante el uso de 2 baterias 18650 de iones de litio, que en valor nominal nos aportan cada una 3.7v y 2600mAh de capacidad. Dado que la construccion que empleamos es serie, la tensión nominal que entregan dichas baterias es de aproximadamente 7.4v y 5200mAh, en vacio aproximadamente 8.4v.

Estas baterias a su vez, soportaran la recarga mediante el empleo de unos celulas fotovoltaicas, que en su construccion aportan 9v de tensión, en vacio 10.25v aproximadamente, y 200mA de corriente maxima teorica, 150mA de corriente real.

La carga de dichas baterias se realiza regulando la tensión que aportan las celulas fotovoltaicas mediante un modulo de carga adecuado para 2 baterias 18650 en topologia serie, el cual regula tanto la sobrecarga, la subdescarga y la corriente maxima de descarga de estas.

El modulo anterior de carga, se conecta a la PCB creada para distribuir las tensiones necesarias para la alimentacion de la placa STM32 y de los sensores que empleamos, dividiendo la tensión de entrada (8.4v) en dos railes de 5v y uno de 3.3v, debido a la necesidad según los diferentes sensores que empleamos.

Para la regulación de 5V se ha empleado directamente el C.I LM7805 sin necesidad de añadir electrónica adicional puesto que la tensión de Drop-out del regulador es de 2V y esto no presenta un problema al tener margen a partir de los 7.4 V de entrada. Se ha preferido repartir la corriente que entregan cada uno de los reguladores en lugar de emplear un solo regulador para no degradar las prestaciones que estos ofrecen en base a la corriente Io que entregan. Por otro lado, la regulación a 3.3 V si ha requerido de electrónica adicional para lograr dicha tensión de salida a partir de los 5V que ofrece el LM7805, de tal modo que se han incorporado una resistencia fija de 240 Ω a la salida del regulador junto con un potenciómetro de valor nominal 1kΩ entre el pin de ajuste y la resistencia fija con el fin de lograr una tensión mas exacta de 3.3V alcanzando un valor resistivo en el potenciómetro de 390 Ω y aprovechando la corriente que circula por ella de aproximadamente 5mA . La ecuación por La que se obtienen los 3.3 V es la siguiente : $Vo = 1.25V \cdot (1 + R2/R1) + R2 \cdot Iadj$, donde si sustituimos obtenemos que :

$Vo = 1.25 \cdot (1 + 390/240) + 390 \cdot 5mA = 3.28$ V, en la práctica ajustando el potenciómetro se obtienen los 3.3V exactos.

Tras las pruebas realizadas, el consumo que se muestra en las baterias sin tener conectadas las celulas de carga fotovoltaicas es de aproximadamente 350mA en total. Según este dato, las baterias serian capaces de aguantar desde su carga maxima (8.4v) hasta el nivel que soporta el modulo de carga de bajotension (7.2v) y con su valor de capacidad (5200mAh) haciendo uso de una corriente de 350mA aproximadamente, en total un tiempo de 14 horas de manera teorica, pero bien sabemos que este valor en la practica se quedara alrededor de 10 horas. Si los modulos

fotovoltaicos estan suminstrando toda la tensión que pueden a su valor de corriente maximo, el consumo de las baterias se reduciria a 200mA (350mA -150mA), por lo que la vida util de una carga de las baterias acenderia a 26 horas de manera teorica, lo que nos acerca a unas 23 horas practicas.

3.3 Sensores integrados (SPI, I2C, etc)

Módulo del sensor PIR:

El sensor PIR tiene 2 **modos de funcionamiento**: 1 solo disparo o disparos periodicos. Para el uso de la aplicación es mejor **el modo de disparos periodicos**.

A su vez, es posible configurar 2 parámetros con los potenciometros, uno para ajustar la distancia (3 metros a 7 metros) y otro para ajusta el tiempo de alarma activa (3 segundos a 5 minutos). Sin embargo, el ajuste del potenciometro que implica la variacion de la distancia de activacion del sensor es indiferente para nuestra aplicación, sin embargo el ajuste sobre el potenciometro que regula el tiempo de activacion de este sistema si que aparenta ser mas critico, debido a que si establecemos un tiempo de activacion muy alto (5 minutos) el sistema puede parecer que no este funcionando. Por el contrario, si el tiempo de activacion es muy corto (3 segundos) el sistema de deteccion puede parecer que funciona de manera incorrecta, dando lugar a posibles equivocos junto al codigo de la aplicación. De esta manera, el ajuste sobre este potenciometro debe de realizarse de manera "fina" entendiendo siempre que tipo de desarrollo queremos para nuestra aplicación.

Módulo del sensor de temperatura y humedad:

El sensor de temperatura y humedad es un **SHT30** con dirección **0x45**. Asimismo, hay diferentes configuraciones, pero el consumo del sensor es muy reducido en sus **2 modos de funcionamiento** (Periodic Data Adquisition y Single Shot).

Parameter	Symbol	Condition	Min.	Typ.	Max.	Units	Comments
Supply voltage	V_{DD}		2.4	3.3	5.5	V	
Power-up/down level	V_{POR}		2.1	2.3	2.4	V	
Slew rate change of the supply voltage	$V_{DD,slew}$		-	-	20	V/ms	Voltage changes on the VDD line between $V_{DD,min}$ and $V_{DD,max}$ should be slower than the maximum slew rate; faster slew rates may lead to reset;
Supply current	I_{DD}	idle state (single shot mode)	-	0.2	2.0	µA	Current when sensor is not performing a measurement during single shot mode
		idle state (periodic data acquisition mode)	-	45	70	µA	Current when sensor is not performing a measurement during periodic data acquisition mode
		Measuring	-	800	1500	µA	Current consumption while sensor is measuring
		Average	-	2	-	µA	Current consumption (operation with one measurement per second at lowest repeatability, single shot mode)

Figura 3: Consumo en sus modos de funcionamiento

Debido que el modo **Periodic Data Adquisition** solo es necesario configurarlo una sola vez se usa ese modo de funcionamiento. Cuando se leen los registros de temperatura y humedad relativa es necesario realizar una conversión para obtener los valores reales de temperatura y humedad relativa. Esas conversiones corresponden a las fórmulas de la siguiente figura:

Relative humidity conversion formula (result in %RH):

$$RH = 100 \cdot \frac{S_{RH}}{2^{16} - 1}$$

Temperature conversion formula (result in °C & °F):

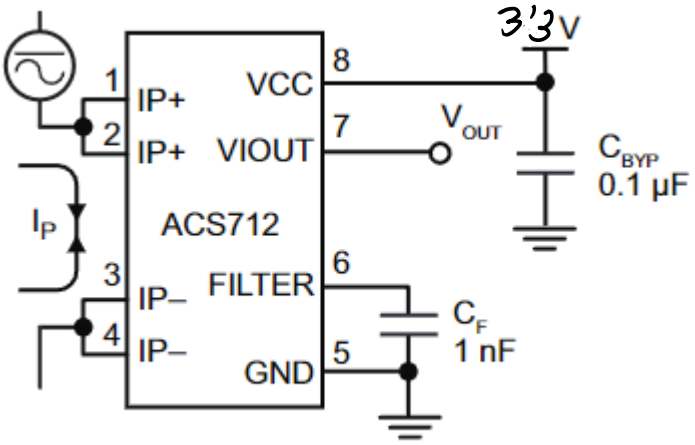
$$T[°C] = -45 + 175 \cdot \frac{S_T}{2^{16} - 1}$$

$$T[°F] = -49 + 315 \cdot \frac{S_T}{2^{16} - 1}$$

Figura 4: Conversión de la temperatura y la humedad relativa

Módulo del sensor de corriente (ADC):

El modulo del sensor de corriente ha sido adquirido a traves de internet y presenta el siguiente esquema:



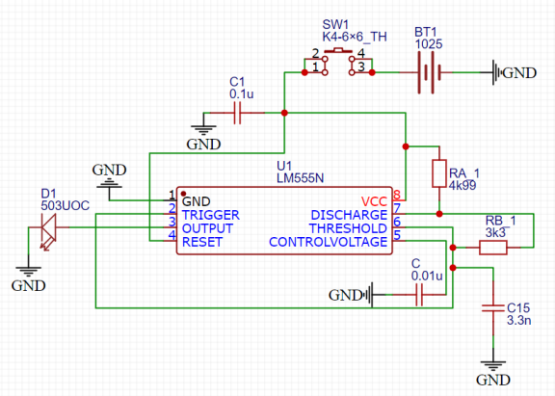
Con el uso de este esquema, conseguimos una ganancia de 185mV/A en la salida analogica Vout. El sensor funciona entregando una tensión a su salida que equivale a la mitad de la tensión de alimentacion mas la tensión transformada de la corriente leida:

$$V_{out}(V) = \frac{3.3(V)}{2} + (I_p(A) * 0.185 \frac{V}{A})$$

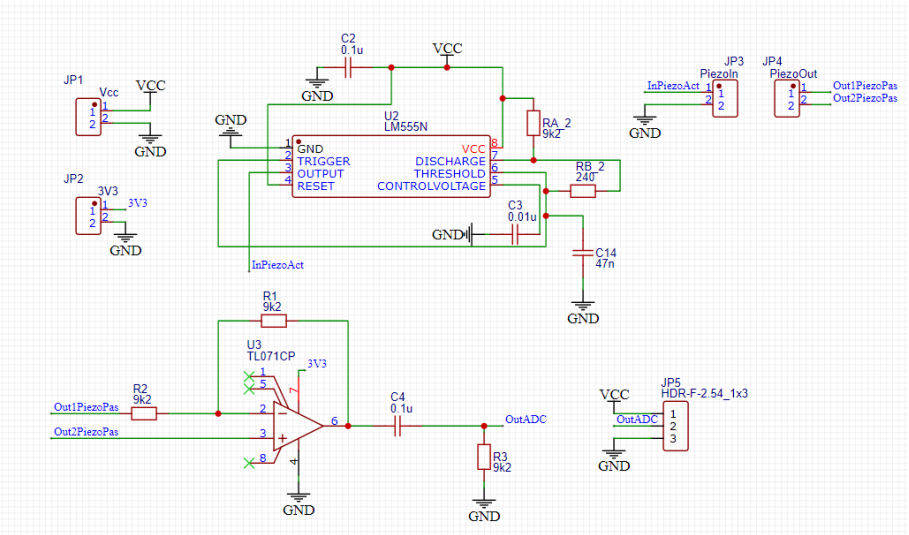
Esto desemboca en una tensión que oscila entre 1.67v (Ip = 0 A) y 1.74v (Ip = 350mA). Despues dicha tensión sera introducida por uno de los pines analogicos de la placa STM32 para su lectura y posterior transformacion en corriente, aplicando la formula de arriba despejando la corriente Ip:

$$I_p(A) = (V_{out}(V) - \frac{3.3(V)}{2})/0.185\frac{V}{A}$$

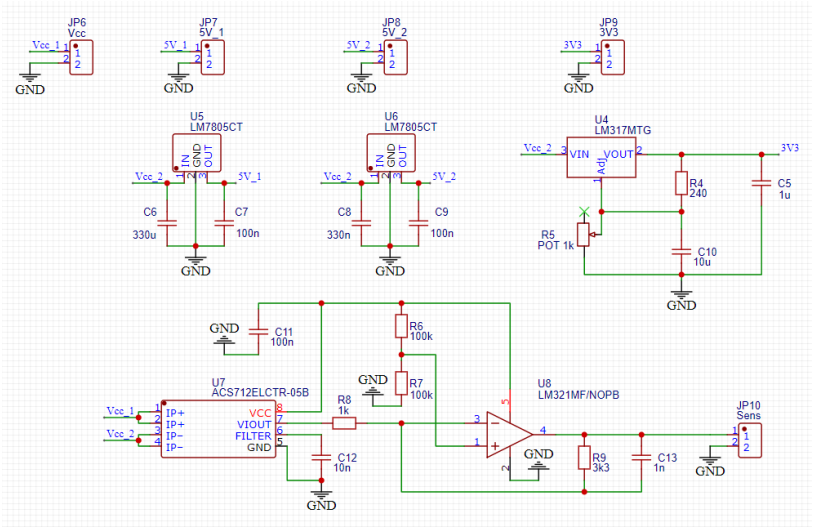
4 DESARROLLO DE SUBSISTEMAS



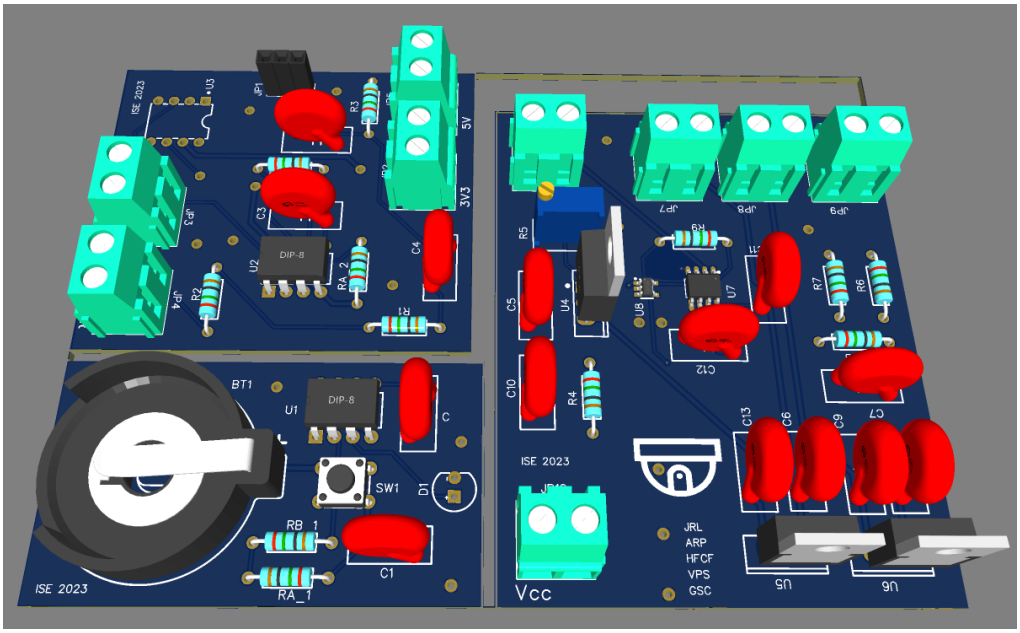
PCB mando a distancia



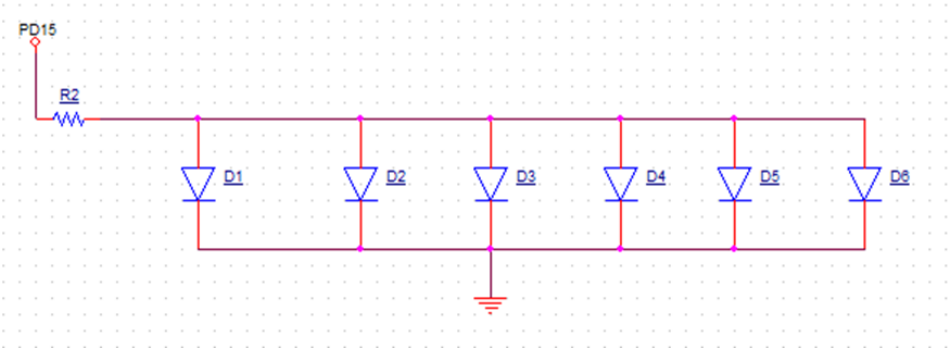
PCB sensor piezoelectrico, conectamos la salida OutADC al pin PA3



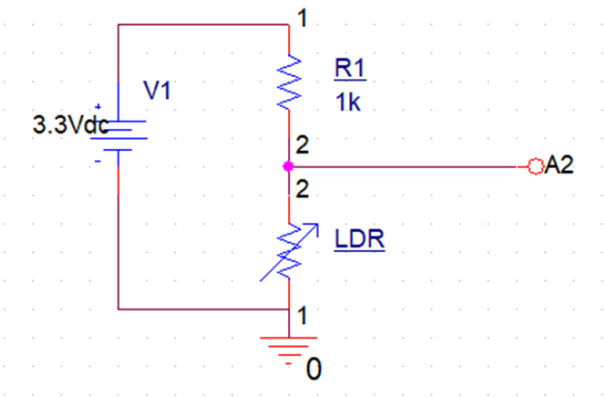
PCB banco de alimentacion, con una de las alimentaciones de 5v alimentaremos la placa STM32, ademas de varios de los sensores. Con la alimentacion restante de 5v y la de 3.3v seguiremos alimentando sensores



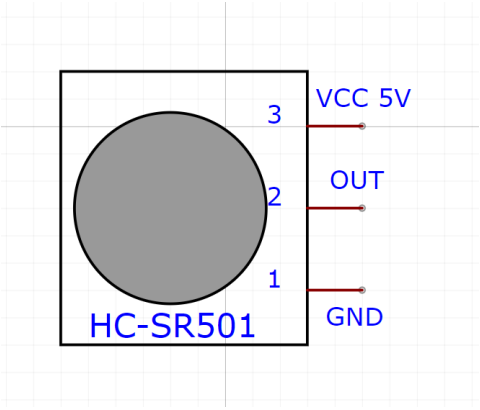
Simulacion 3D de las 3 PCBs diseñadas



Conexiones de los leds, estos iran conectados a la placa STM32 mediante el pin PD15

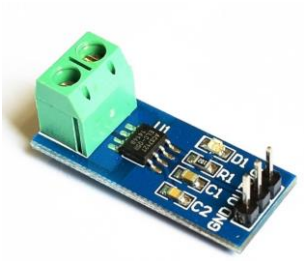


Conexión de la LDR, dicha salida ira conectada al pin PC3 de la STM32

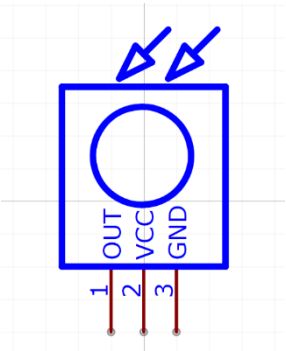


Esquema del sensor PIR (HC-SR501), la salida OUT del sensor ira conectada al pin PE11 de la placa STM32

Sensor de temperatura y humedad (SHT30), las conexiones a la placa seran: pin SCL al PB8, pin SDA al PB9, pin 3V3 a la alimentacion de 3.3v y GND a la masa general de la alimentación.



Sensor de corriente (ACS712), el conexionado de este sensor sera: pin Vcc a 3.3v, GND a la masa general de la alimentacion, pin OUT al PC0. Por otro lado en el bornero verde conectaremos la alimentacion directamente desde las baterias como si de un amperimetro se tratase.

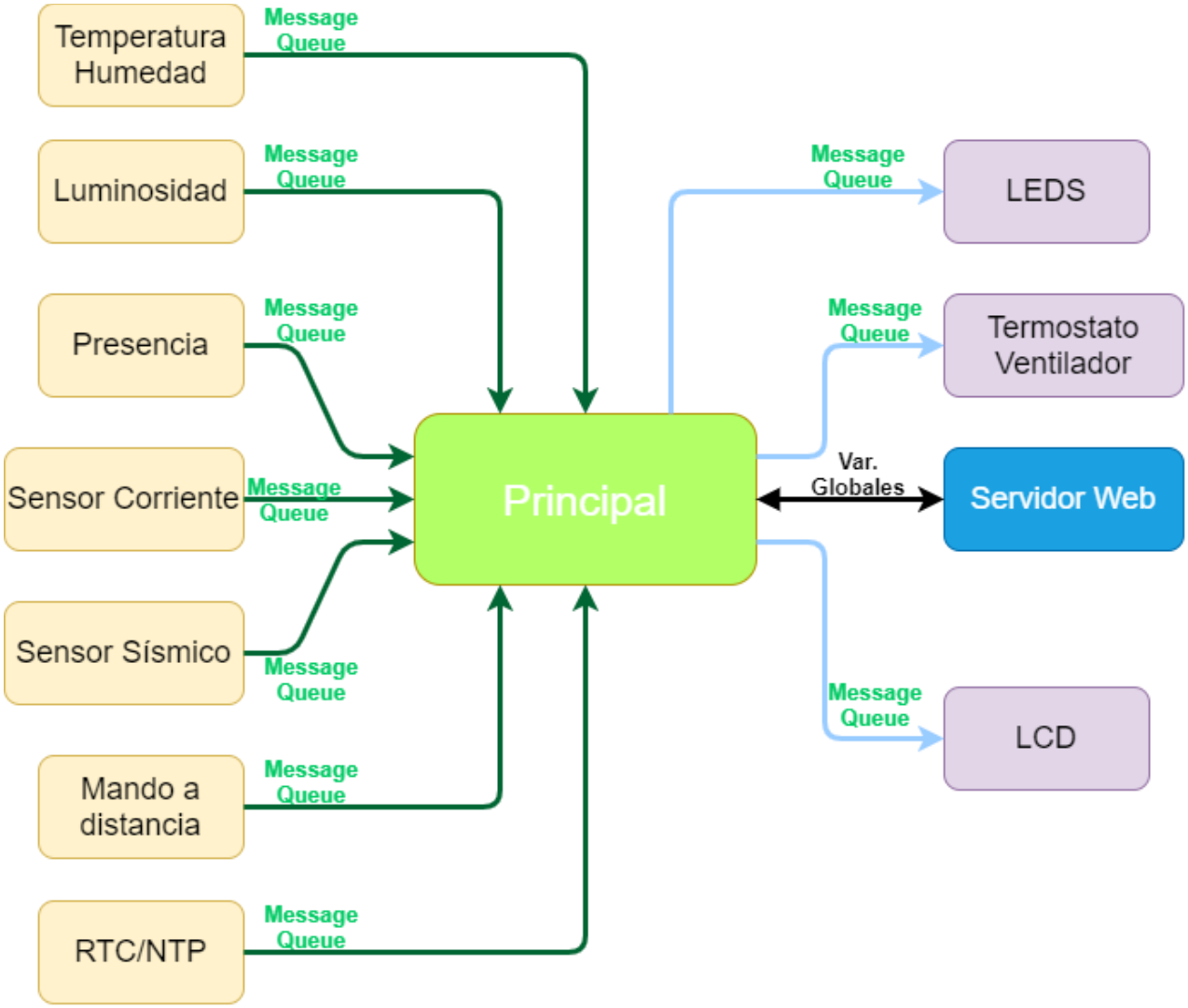


Receptor Infrarrojo (TSOP2238), dicho receptor utiliazado para recibir la señal del mando a distancia, se conectara de esta manera: pin OUT al PD12, Vcc a 5v y GND a la masa general del circuito.

Ventilador utilizado para el termostato de la casa.

5 SOFTWARE

5.1 Descripción de cada uno de los módulos del sistema



Módulo del sensor de temperatura y humedad:

Este módulo configura el sensor de temperatura y humedad relativa que funciona con el protocolo de comunicación I2C. Se configura en su modo de adquisición periodica para luego leer los datos de los registros de lectura. Tras eso, se realiza una conversión de los datos obtenidos, se envía a una cola de mensajes y espera 1 segundo para volver a leer los registros, esto se realiza sucesivamente.

Módulo del LCD:

Este módulo configura inicialmente el protocolo de comunicación SPI para poder usar el LCD, tras esa configuración inicial se inicializa una cola donde a través de una estructura guardará la

información de la temperatura, la humedad relativa, las horas y la fecha. Asimismo, con la pulsación del pulsador azul se muestra en el LCD el consumo de la aplicación.
Tras la captura de los datos en el display mostrará la información obtenida de la cola de mensajes y refrescará cada 1 segundo.

Módulo del sensor PIR:

Este módulo gestiona las interrupciones generadas por el sensor PIR. Cuando hay una interrupción se envía por una cola de mensajes si se ha detectado algún movimiento.

Módulo del mando a distancia:

Este módulo gestiona la activación de una señal de habilitación para el módulo del garaje. Está señal tendrá dos posibles valores, nivel alto y nivel bajo, dependiendo de si el receptor LED, recibe información del emisor a través de una frecuencia determinada, en nuestro caso, alrededor de 38 KHz.

Módulo del RTC:

Este módulo configura el RTC. Cada que pasa 1 segundo aumenta los segundos en una unidad, simulando de esta forma un reloj.

Módulo del SNTP:

Este módulo es el encargado de sincronizar el RTC con el tiempo real usando el protocolo SNTP.

Módulo del LDR:

Este módulo es el encargado de procesar la luz que hay en el ambiente mediante la lectura de la tensión que cae en el LDR. En funcion de la luz recibida, graduaremos el nivel de brillo que emiten los leds de la casa con el manejo de una señal PWM.

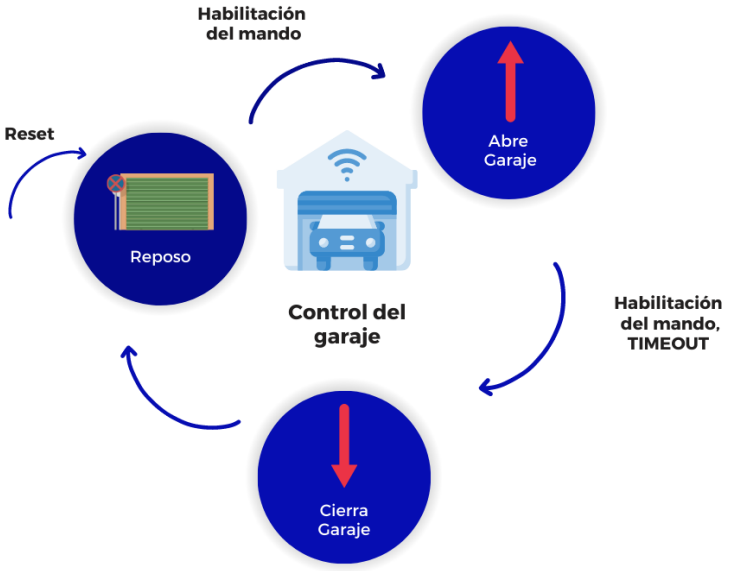
Módulo del Termostato:

Este módulo es el encargado de encender o apagar un ventilador en función de un determinado umbral. Cuando la temperatura del ambiente es mayor que el umbral, se encenderá el ventilador. De lo contrario, se apagará.

Módulo del Garaje:

Una vez se ha recibido la señal de habilitación del módulo del mando a distancia, este bloque se encargar de manejar con un pwm la activación de un motor que abra o cierra la puerta de un garaje. Por defecto la puerta estará cerrada, ya que en caso de solo detectar una activación por parte del módulo mando a distancia (abrir), se cerrará sola con un cierto TIMEOUT. Después de la primera

habilitación del módulo mando a distancia, sucesivas activaciones harán variar el estado de la puerta del garaje entre subir o bajar.



Módulo del bajo consumo:

A través de la pagina web, podremos solicitar entrar en el modo sleep del microprocesador, con el objetivo de reducir el consumo. Nos mantendremos dentro de este modo hasta que se produzca una interrupción como la que puede generar el boton de usuario de la placa o una pulsacion del mando del garaje. Por otra parte, antes de entrar al modo sleep, guardaremos las ultimas 100 medidas de temperatura y humedad que se hayan realizado junto a su time stamp dentro de la memoria flash.

Módulo del sensor sismo:

A través del pin PA3 del ADC, el microprocesador va a recibir la información de la señal del piezoeléctrico. La señal de salida de los piezoelécricos, después de usar el osciloscopio, podemos ver que tiene una duración de unos 25 ms. Para disminuir el consumo del proyecto y mejorar la eficiencia del programa, tenedremos un hilo que podrá a funcionar el ADC, cada 70 ms a través de un timer virtual. De esta manera, y teniendo en cuenta que un terremoto tiene una cierta duración y no es algo puntual, el microprocesador detectará en alguna de sus medidas cada 70 ms, una señal mayor que 0.5v, y activa el LED verde de la STM32, mostrando así, que ha sido detectada una vibración en el suelo de la casa.

Módulo del sensor de corriente:

Las pruebas realizadas sobre el modulo del sensor de corriente tienen que ver en primera instancia con la correcta lectura de la tensión que este aporta. Por lo qu euna vez conectadas las baterias al modulo de sensado de corriente, se conecta la salida de este a un pin analogico y se configura un ADC de la placa para leer tensión entre 0 y 3.3v. Las pruebas resultaron exitosas en esta lectura usando una resolucion de 12 bits y aplicando un filtro software que hace la media de 50 medidas tomadas, para evitar el ruido de la corriente en la lectura.

Despues se procede a transformar dicha tensión en la corriente que necesitamos aplicando esta formula:

$$I_p(A) = \left(V_{out}(V) - \frac{3.3(V)}{2}\right)/0.185\frac{V}{A}$$

Y comprobamos que para un consumo alto (<200mA) la lectura es muy precisa, mientras que para consumos bajos (>100mA) la lectura fluctua mucho y se puede confundir.

Módulo LEDS:

Este módulo es el encargado de gestionar el encendido y apagado de los LEDs utilizados. Gracias a la información recolectada por otros módulos como el del PIR, LDR y el detector de terremotos.

5.2 Descripción global del funcionamiento de la aplicación. Descripción del autómata y del diagrama con el comportamiento del software

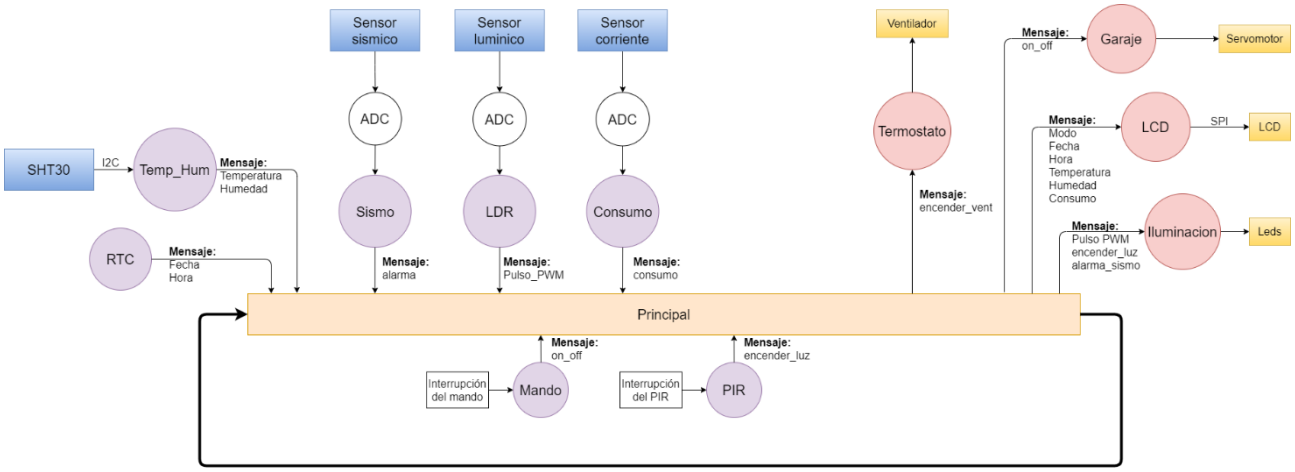


Figura 5: Diagrama de funcionamiento del hilo Principal

El funcionamiento de la aplicación es la siguiente:

Inicialmente, tras inicializar toda los hilos, las colas de mensajes y otros recursos, se obtiene información de los módulos de entrada del sistema. Esta información se envía a los módulos de salida, ya sea para representar información, encender las luces, abrir la puerta de un garaje, etc

La información que envía los módulos Temp_Hum y RTC son los representados en el LCD. Asimismo, el valor de la temperatura es usada para gestionar el encendido de ventilador en el hilo ThTermostato.

En los módulos Sismo, LDR y Consumo se obtiene información a través de los ADC de la Tarjeta Nucleo STM32F429Zi, esta información se envía al hilo Principal.

La información del LDR y de la interrupción del PIR son usados para gestionar la intensidad de iluminación y encendido de las luces de la casa. La información del Sismo sirve para encender una el led rojo de la placa, indicando que ha ocurrido un terremoto.

La interrupción del mando es la encargada de abrir o cerrar las puertas del garaje.

Además, la aplicación incluye un servidor web embebido, donde se muestra información y se puede realizar interraciones.

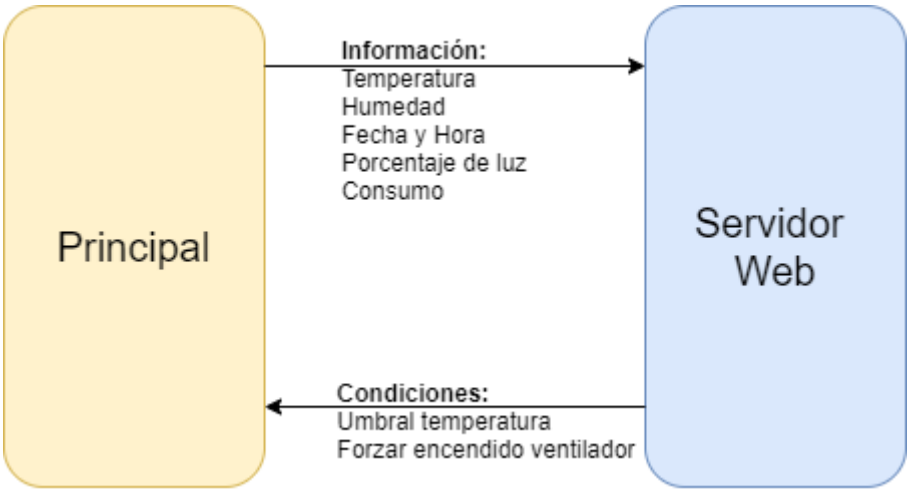


Figura 6 : Interacción entre el Principal y el Servidor Web

En el hilo Principal se modifica la información de la temperatura, humedad, fecha, hora, porcentaje de luz y consumo, todos estoy datos están como variables globales y se envía al servidor web. La información enviada no se puede modificar desde otros ficheros, solamente desde el Principal.

En el servidor web se puede realizar interacciones desde la pestaña Ventilador, donde se puede modificar el umbral de temperatura, el tipo de funcionamiento (Manual o automático) o forzar el encendido o el apagado del ventilador si está en modo manual, de forma que se puede el ventilador. Esta información lo recibe el principal como condiciones que se usan para interactuar con el módulo ThTermostato.

5.3 **Descripción de las rutinas más significativas que ha implementado.**

Módulo	Funciones	Otras funciones que se llama	Más detalles
Principal.c	Init_ThPrincipal	osThreadNew()	Inicia el hilo principal.
	ThPrincipal	osMessageQueueGet() osMessageQueuePut()	Gestiona las colas de mensajes de diferentes módulos. Más información: **Tabla 4: Entradas y Salidas del Principal**
	Init_All_Pins	Init_PIR_Pin() Init_PWM_Pin() Init_Mando_Pin()	Inicia todos los pines de la aplicación
	Init_All_Threads	Init_ThThermostato() Init_ThLCD() Init_ThRTC() Init_ThSNTP() Init_ThTemp_Hum() Init_ThPIR() Init_ThLDR() Init_ThIluminacion() Init_ThMando()	Inicia todos los hilos de la aplicación
Lcd.c	Init_ThLCD	LCD_reset() LCD_Init()	Inicia el hilo del lcd.c y configura la comunicación SPI
	ThLCD	Init_MsgLCD() LCD_clear() LCD_update() osMessageQueueGet() Sprintf() LCD_symbolToLocalBuffer_L1() LCD_symbolToLocalBuffer_L2() osThreadYield()	Muestra información sobre la temperatura, la humedad relativa, las horas y la fecha en el LCD en función de los datos que recibe por la cola de mensajes.
	Init_MsgLCD	osMessageQueueNew()	Inicia la cola de mensajes del LCD

	LCD_reset	SPI_Init() HAL_GPIO_Init() HAL_GPIO_WritePin() Reset_Signal()	Inicia el SPI, configura los pines de datos, de reset y de CS, y genera una señal de reset
	LCD_Init	LCD_wr_cmd()	Envía comandos para configurar el LCD
	LCD_update	LCD_wr_cmd() LCD_wr_data()	Actualiza los pixeles del LCD en función del contenido de un buffer de tamaño 512
	LCD_clear	LCD_update()	Pone a 0 todos los elementos del buffer de tamaño 512 y actualiza los pixeles del LCD
	LCD_symbolToLocalBuffer_L1	------	Escribe en las páginas 0 y 1 del LCD
	LCD_symbolToLocalBuffer_L2	------	Escribe en las páginas 2 y 3 del LCD
Temp_Hum.c	Init_ThTemp	osThreadNew() I2C -> Initialize () I2C -> PowerControl () I2C -> Control ();	Inicia el hilo de la temperatura y humedad, y configura la comunicación I2C
	ThTemp	Init_MsgTemp() Init_timer_TempHum() Config_Communication() ReadSensorData() osMessageQueuePut()	Realiza las transacciones de escritura y lectura para obtener la temperatura y la humedad relativa. Se hace una conversión y se envía a la cola de mensajes. Inicia un timer virtual y espera a que se active un flag para volver a obtener valores de tempertura y humedad relativa.
	I2C_SignalEvent_TEMP_HUM	osThreadFlagsSet()	Activa un flag del hilo cuando se ha acabado una transferencia I2C.
	Init_MsgTemp	osMessageQueueNew()	Inicia la cola de mensajes del Temperatura.c
	Init_timer_TempHum	osTimerNew()	Inicia un timer para leer los registros cada 1 segundo
	Timer_TempHum_Callback	osThreadFlagsSet()	Callback del timer virtual, activa el flag 0x2 para indicar que puede volver a leer los registros del sensor.
	Config_Communication	Set_PeriodicAdq_Mode() SetSingleShotMode() I2C -> MasterTransmit () osThreadFlagsWait()	Elige el modo de adquisición de datos: 1 es para el modo periodico y 0 para obtener datos 1 vez. Realiza una transacción de escritura para configurar los regitros del sensor.
	Set_PeriodicAdq_Mode	------	Configura los comandos de escritura para que se modifiquen los registros de lectura periodicamente.

	SetSingleShotMode	------	Configura los comandos de escritura para que se modifiquen los registros de lectura una vez.
	ReadSensorData	I2C -> MasterReceive () osThreadFlagsWait()	Lee los registros del sensor, realiza la conversión de los datos y la información que se va a intrudir en la cola de mensajes.
PIR.c	Init_PIR_Pin	__HAL_RCC_GPIOE_CLK_ENABLE() HAL_GPIO_Init() HAL_NVIC_EnableIRQ()	Inicializa y habilita las interrupciones del PIN E11 tanto la bajada como la subida.
	Init_MsgPIR	osMessageQueueNew()	Inicializa la cola de mensajes del PIR
	EXTI15_10_IRQHandler	HAL_GPIO_EXTI_IRQHandler()	Maneja las solicitud de las interrupciones externas del pin 11
	HAL_GPIO_EXTI_Callback	HAL_GPIO_ReadPin() osMessageQueuePut()	Vigila si el pin está a nivel alto o bajo después de la interrupción, en función de eso indica si ha habido movimiento o no y lo envía a la cola de mensajes.
SNTP.c	Init_ThSNTP	osThreadNew()	Inicia el hilo del SNTP
	SNTP_Thread	Init_Timer_SNTP() osTimerStart() osThreadFlagWait()	Se encarga de sincronizar el RTC cada 5 minutos e inicializa el timer virtual.
	Init_Timer_SNTP	osTimerNew()	Función que inicializa el timer virtual.
	TimerSNTP_Callback	Get_SNTP_Time() osThreadFlagsSet()	Callback del timer virtual donde se obtiene el tiempo local que se usa para actualizar el RTC
	get_SNTP_Time	netSNTPc_GetTime()	Función encargada de enviar una petición SNTP
	time_callback	getLocalTime	Callback usado en la función netsSNTPc_GetTime, si la función ha devuelto un valor no nulo realiza una conversión de segundos al tiempo local.
	getLocalTime	strftime() *localtime() Set_RTC_Time() Set_RTC_Date()	Función encarga de actualizar el tiempo y la fecha del RTC al tiempo actual.
RTC.c	Init_ThRTC	osThreadNew() RTC_Init()	Inicia el hilo del RTC
	RTC_Thread	Init_timer_RTC() Init_MsgRTC() osTimerStart()	Inicializa el timer virtual y la cola de mensajes del RTC. También se encarga de actualizar el RTC cada 1 segundo

		osThreadFlagsWait()	
	Init_timer_RTC	osTimerNew()	Función que inicializa el timer virtual.
	TimerRTC_Callback	Get_Date_Time() osMessageQueuePut() osThreadFlagsSet()	Callback del timer virtual donde se obtiene el tiempo y la fecha del RTC para enviarlo a una cola de mensajes.
	Init_MsgRTC	osMessageQueueNew()	Inicia la cola de mensajes del RTC.c
	Get_Date_Time	HAL_RTC_GetTime() HAL_RTC_GetDate()	Función que obtiene el tiempo y la fecha (En ese orden).
	RTC_Init	__HAL_RCC_RTC_ENABLE() HAL_PWR_EnableBkUpAccess() __HAL_RCC_PWR_CLK_ENABLE() HAL_NVIC_EnableIRQ() HAL_RTC_Init()	Inicializa y habilita el RTC
	Get_RTC_Time	HAL_RTC_GetTime()	Función que obtiene el tiempo del RTC
	Get_Date_RTC	HAL_RTC_GetDate()	Función que obtiene la fecha del RTC
	Set_RTC_Time	HAL_RTC_SetTime()	Función que configura el tiempo del RTC
	Set_RTC_Date	HAL_RTC_SetDate()	Función que configura la fecha del RTC
LDR.c	Init_ThLDR	osThreadNew() ADC1_pins_F429ZI_config()	Inicia el hilo del LDR y configura los pines del ADC
	ThLDR	Init_MsgLDR() ADC_getVoltage() osMessageQueuePut()	Se encarga de inicializar la cola de mensajes y obtiene la tensión a la salida del LDR. Cuando el porcentaje de la tensión actual es diferente al porcentaje anterior se envía el porcentaje del pulso a la cola de mensajes.
	Init_MsgLDR	osMessageQueueNew()	Inicia la cola de mensajes del LDR.c
Iluminacion.c	Init_ThIluminacion	osThreadNew() Init_PWM_Iluminacion_Pin()	Inicia el hilo de la Iluminacion y configura los pines
	ThIluminacion	Init_MsgIluminacion() osMessageQueueGet() Config_PWM_Pulse()	Se encarga de inicializar la colade mensajes y en función del dato recibido configura el PWM
	Init_PWM_Iluminacion_Pin	__HAL_RCC_GPIOD_CLK_ENABLE()	Función que inicializa y habilita el canal 4 del Timer 4 como PWM.

		__HAL_RCC_TIM4_CLK_ENABLE() HAL_GPIO_Init() HAL_TIM_PWM_ConfigChannel()	
	Config_PWM_Pulse	HAL_TIM_PWM_DeInit() HAL_TIM_PWM_Stop() HAL_TIM_OC_Init() HAL_TIM_PWM_Init() HAL_TIM_PWM_ConfigChannel() HAL_TIM_PWM_Start()	Función que para el pulso del PWM, configura a un nuevo pulso y lo vuelve a iniciar.
	Init_MsgIluminacion	osMessageQueueNew()	Inicia la cola de mensajes del Iluminacion.c
ThTermostato.c	Init_ThThermostato	osThreadNew()	Inicia el hilo de la ThTermostato.c
	ThTermostato	Init_Ventilador_Msg() osMessageQueueGet() encender_ventilador() apagar_ventilador()	Se encarga de inicializar la cola de mensajes y en función del dato recibido enciendo o apaga el ventilador.
	init_ventilador	__HAL_RCC_GPIOC_CLK_ENABLE() HAL_GPIO_Init() HAL_GPIO_WritePin()	Función que inicializa el pin del Ventilador.
	Init_Ventilador_Msg	osMessageQueueNew()	Inicia la cola de mensajes del Termostato
	encender_ventilador	HAL_GPIO_WritePin()	Enciende el ventilador
	apagar_ventilador	HAL_GPIO_WritePin()	Apaga el ventilador
Mando.c	Init_ThMando	osThreadNew()	Inicializa el hilo de Mando.c
	Init_Mando_Pin	__HAL_RCC_GPIOD_CLK_ENABLE() HAL_NVIC_EnableIRQ() HAL_GPIO_Init()	Configura el pin para que se pueda recibir interrupciones del mando
	Init_Timer_Rebotes	osTimerNew()	Inicializa un timer para evitar rebotes
	Init_MsgQueue_Mando	osMessageQueueNew()	Inicializa la cola de mensajes
	ThMando	Init_MsgQueue_Mando()	Se encarga de inicializar el timer virtual y la cola de mensajes. Cuan hay una interrupción se arranca el timer virtual y se envía mensajes a la cola de mensajes.

		Init_Timer_Rebotes() osThreadFlagsWait() osTimerStart()	
Consumo.c	Init_ThConsumo	osThreadNew()	Inicializa el hilo de Consumo.c
	Init_MsgConsumo	osMessageQueueNew()	Inicializa la cola de mensajes
	timer_Consumo_Callback	osThreadFlagsSet()	Callback del timer virtual usado para tomar muestras de la corriente cada x tiempo
	Init_timer_Consumo	osTimerNew	Inicializa un timer virtual
	Consumo_Thread	Init_MsgConsumo () Init_timer_Consumo () ADC1_pins_F429ZI_config() ADC_Init_Single_Conversion() ADC_getVoltage() osThreadFlagsWait() osTimerStart() osMessageQueuePut()	Inicialmente inicializar el timer virtual, la cola de mensajes y los pines del ADC. Tras la configuración se toma 50 muestras del ADC, se hace la media y se corrige el error de la medida. Finalmente, se envia un mensaje a la cola de mensajes.
Garaje.c	Init_ThGaraje	osThreadNew() Init_PWM_Garaje	Inicializa el hilo de garaje.c
	Init_PWM_Garaje	__HAL_RCC_GPIOD_CLK_ENABLE() __HAL_RCC_TIM4_CLK_ENABLE() HAL_TIM_PWM_ConfigChannel ()	Configura el timer harware usado para mover la puerta del garaje con un servomotor
	ThGaraje	Init_MsgQueue_Garaje osMessageQueuGet	Se encarga de inicializar la cola de mensajes y en función del dato recibido abre o cierra el garaje.
	Init_MsgQueue_Garaje	osMessageQueueNew()	Inicia la colad e mensajes
	Config_PWM_Pulse_Garaje	HAL_TIM_PWM_DeInit() HAL_TIM_PWM_Stop() HAL_TIM_OC_Init() HAL_TIM_PWM_Init() HAL_TIM_PWM_Start()	Funcion usada para mover la puerta del garaje

Tabla 3: Rutinas de los módulos

Tabla de mensajes del Principal:

Mensajes		
Entradas	**Acciones**	**Salidas**
mid_MsgPIR	Lleva el dato de encender las luces de la casa cuando hay movimiento	mid_MsgIluminacion
mid_MsgRTC	Lleva los valores de las horas y fechas que se van a mostrar en el LCD y la web	mid_MsgLCD
mid_MsgLDR	Lleva los valores de la intensisdad de la luz	mid_MsgIluminacion
mid_MsgTemp_Hum	Lleva los valores de la temperatura y humedad que se van a mostrar en el LCD y en el servidor web	mid_MsgLCD
mid_MsgMando	Lleva el dato que abre y cierra la puerta del garaje	mid_MsgGaraje
mid_MsgSismo	Lleva el dato que indica que ha habido un sismo	mid_MsgIluminacion
mid_MsgConsumo	Lleva el valor del consumo	---

Tabla 4: Entradas y Salidas del Principal

6 DEPURACION Y TEST

6.1 **Pruebas realizadas software.**

Prueba del sensor de temperatura y humedad:

La prueba del sensor de temperatura y humedad no se necesitan condiciones de entrada.

Esta prueba consiste en verificar el funcionamiento del módulo Temp_Hum.c, tales como los valores correspondientes de las medidas y el correcto funcionamiento de la cola de mensajes. Asimismo, sólo se necesita un módulo de test para comprobar el funcionamiento de la cola de mensajes y con las herramientas, Debugguer y Debug printf, ofrecidas por el entorno de desarrollo Keil permite observar los printf de forma que se puede comprobar los valores devueltos por el sensor.

Se espera que el módulo Temp_Hum cuando realice una medida cada 1 segundo para enviar los resultados a la cola de mensajes y con la ayuda de otro módulo, Test, extrae los resultados y los muestra los resultados obtenidos con un printf.

Los resultados obtenidos son los esperados, cuando se obtiene información del sensor esta se envía a la cola de mensajes y el módulo de test extrae la información y la representa en el terminal.

Prueba del sensor PIR:

La prueba del sensor PIR necesita condiciones de entrada, esta es la salida del sensor que puede ser un nivel alto o un nivel bajo.

Esta prueba consiste en verificar el funcionamiento del módulo PIR, se requiere de un módulo de leds porque el sensor PIR es la encargada de encender o apagar la iluminación de la casa.

Se espera que cuando haya algún movimiento se debe encender un led, mientras que cuando no haya movimiento el led debe estar apagado.

Los resultados obtenidos son los esperados porque cuando hay movimiento en frente del sensor PIR detecta el movimiento el led se enciende cuando ha dejado de haber moviemiento el led se apaga.

Prueba del sensor lumínico:

La prueba del sensor luminico no necesita condiciones de entrada.

Esta prueba consiste en verificar el funcionamiento del módulo LDR, se observan valores de tensión, con la ayuda de los watches que ofrece el entorno de desarrollo Keil uVision, que se obtienen a través del ADC de la tarjeta NUCLEO STM32.

Se espera que haya variaciones de tensión si se tapa el LDR o se ilumina el LDR.

Los resultados obtenidos son los esperados porque hay variaciones de tensión cuando se ilumina o se tapa el LDR.

Prueba del LCD:

La prueba del LCD requiere de los siguientes módulos externos: Temp_Humc y RTC.c

La prueba consiste en comprobar la representación de la temperatura, la humedad relativa, las horas y la fecha en el LCD y el funcionamiento correcto de la cola de mensajes porque así es como este módulo va a obtener la informacion, por ello, se incluye un módulo Test que es la encargada de enviar la información a la cola de mensajes del módulo LCD. Con la pulsación del pulsador azul pasa a mostrar el valor del consumo.

Se espera que los valores de la temperatura, la humedad relativa, el consumo, las horas y la fecha varíen cada segundo (Por lo menos debe variar las horas) en el LCD y que haya ningún problema con las colas de mensajes.

Los resultados son los esperados porque hay cambios en los valores de la información que representa el LCD cada 1 segundo. Además, cuando se pulsa el botón azul muestra el valor del consumo.

Prueba del termostato:

La prueba del termostato consiste en verificar que cuando se supera un cierto umbral de temperatura, el ventilador se encienda. De lo contrario siempre debe de estar apagado. Para verificar esta condición utilizamos diferentes herramientas de depuración como los breakpoints y los watches. Los breakpoints los utilizamos para determinar que cuando se superaba el umbral, el programa entraba a la zona del codigo donde se encendia el ventilador. De lo contrario, entraba a la parte del código donde se apaga el ventilador. Además, la utilización de los breakpoints fueron utilizados para ver cual era la temperatura ambiente en un momento dado.

Prueba de la memoria flash:

La prueba de la memoria flash consiste en verificar que la información que se quiere almacenar a partir de una dirección de la memoria se guarda correctamente. Para ello utilizamos la ventana de depuración "memory". A traves de esta ventana, podemos buscar la información guardada en una cierta posicion de memoria.
Posteriormente, una vez guardada la información a partir de la dirección de memoria deseada. Procedemos a leer los datos guardados con ayuda de la función "lectura_flash_string_format(uint32_t inicio_direccion_escritura, int palabras_lectura)" . Con esta función vamos leyendo palabra a palabra de 32 bits, la información guardada en la memoria flash desde la dirección de memoria incial a partir la cual se han ido guardado los datos.
Por otra parte para la depuración del código que gestiona las escrituras y lecturas de la memoria flash. Hemos utilizado otras herramientas de depuración como los whatches, breakpoints y la ventana de visualización de los printf para detectar posibles errores a la hora de escribir o leer en la memoria flash del microprocesador.

Prueba del bajo consumo:

La prueba consiste en verificar que se entra y que se sale del modo sleep. Para comprobar esta especificación, nos ayudamos del led azul de la placa STM32F429ZI. Cada vez que se entra o que se sale del modo sleep, cambiamos el estado del led. De esta manera, nos aseguramos de que el módulo funciona de la forma deseada. Además, utilizamos puntos de ruptura para ver que el código se esta ejecutando de la manera deseada en todo momento antes de entrar al modo sleep.

Prueba del los LEDs de la casa:

Después de que los modulos del LDR y del PIR funcionaban de la forma correcta. Procedimos interconectar ambos modulos mediante una cola de mensajes. Para comprobar que funcionaban de manera conjunta. Para ello utilizamos puntos de ruptuta, printf, whatches para depurar el código.

Prueba de los servomotores del garaje:

El servomotor se controla mediante posición, regulandose esta en función del ancho del PWM de entrada, de tal modo que una posición de -90º corresponde a un valor a nivel alto del PWM de 1 ms, de los 20 ms de periodo de la señal(50 Hz), mientras que una posición de 90º corresponde a un valor a nivel alto del PWM de 1,5 ms. En resumen, la posición de -90º corresponde a un ciclo de trabajo del 5% del

PWM mientras que una una posición de 90º corresponde a un ciclo de trabajo del 10%. Se probó a ajustar dicho servomotor entre este abanico de tal modo que , a un valor del parámetro pulse del PWM de 1500 sobre el máximo de 8000 la puerta del garaje estuviese en la posición alta(abierta), mientras que para un valor de pulse de 2000 la puerta del garaje se encontrase en la posición baja(cerrada).

6.2 **Pruebas realizadas hardware**

Prueba de los LEDs de la casa:

Utilizamos el multímetro para ver la corriente media en la entrada del pin digital donde estan conectados los LEDs. Con el fin de determinar el consumo maximo del módulo de los LEDs.

Prueba deL módulo LDR:

Utilizamos el multímetro para ver la corriente máxima que circulaba por el LDR. Con el objetivo de calcular su consumo máximo.

Prueba de la flash:

Desconectamos la alimentacion de la placa una vez guardada la información en la flash. Con el objetivo de ver que se habia guardado correctamente la información . Para ello, volvimos a conectar la alimentación de la placa y vimos en la direccion de memoria donde habiamos guardado la información, los datos guardados.

Prueba del bajo consumo:

Empleamos un multímetro, conectado entre las baterías que proporcionan alimentación al sistema construido. Para ver la corriente que estamos consumiendo en en el modo sleep y en el modo normal de funcionamiento.

Prueba del termostato:

Utilizamos el multímetro del laboratorio para verificar que la corriente que atravesaba al ventilador era la deseada cuando este estaba encendido.

Prueba del sensor sísmico:

Durante el desarrollo del proyecto, se incluyó una de las PCBs anteriormente mecionadas. Su función es la de simular una vibración de alrededor de 3KHz, frecuencia de oscilación fundamental del piezoeléctrico. A continuación, se gestionará, a través de otro piezoeléctrico la vibración recibida a través de este primer piezo. El problema, es que la señal recibida, respeta la frecuencia de emisión, pero sin embargo, la amplitud de la señal no super los 300 mV y por tanto será necesario implementar un circuito integrado, 1N4001, como amplificador operacional en modo no inversor con una ganancia de 2V/V. De esta manera, hemos podido trabajar con el piezoeléctrico sin necesidad de estar ejerciendo pequeños golpes para comprobar la funcionalidad del módulo.

Prueba de alimentacion:

Para el banco de alimentaciones se han planteado tres circuitos que cumplen la función de reguladores lineales. Dos de ellos tienen la misma funcionalidad y características, que es la de obtener una alimentación de 5V, y el tercero se encarga de obtener una tensión de 3.3V a la salida.

Despues de soldar todos los componentes de la PCB designada a la alimentacion, se han hecho pruebas y se ha obtenido que el banco de alimentaciones entrega un valor exacto de 4.98v en bornas de la alimentacion diseñada para 5v y 3.3v exactos en bornas de la alimentacion diseñada para 3.3v dado que podemos actuar sobre el potenciometro que regula dicha tensión y alcanzar el nivel deseado.

Prueba del sensor de corriente:

Las pruebas realizadas al modulo de sensado de corriente, en la parte hardware, se corresponden con la correcta lectura de la corriente por parte del ADC de la placa STM32 mientras conectamos y deconectamos modulos que ofrecen consumo a las baterias. Quedo probado que el sensor ofrece una tensión muy precisa de la corriente (transformada en tensión) siempre que conectamos y desconectamos modulos, el problema reside en la transformacion de esta tensión en corriente (parte explicada en “pruebas software”). Con estas pruebas conseguimos deducir la tensión umbral que el sensor entiende por corriente de entrada = 0, resultando en un valor entre 1.669v y 1.67v de salida, mientras que la tensión de salida maxima alcanzada por el sensor de corriente, cuando tenemos conectados todos los modulos a las baterias es de 1.73v (aproximadamente 350mA).

Tambien aprovechamos estas pruebas para concluir si la tensión que debia alimentar este modulo era de 3.3v o de 5v. Finalmente se llego a la conclusion de que al alimentar el modulo con 3.3v las medidas contenian menos ruido y eran mas sencillas de tomar.

7 CONCLUSIONES

Con la finalización del proyecto, cada uno de los miembros del equipo va a dar su propia conclusión.

HAO FENG: Como resultado del trabajo grupal de la asignatura, se ha llevado a cabo un proyecto de mediana complejidad con sus correspondientes subsistemas analógicos y digitales, donde la integración final ha dado lugar la aplicación de una casa domotizada con diferentes funcionalidades que cumplen con los objetivos de la asignatura. Durante la realización del proyecto se han aplicado los conocimientos que se han adquirido previamente a lo largo del grado universitario, demostrando así su importancia. Para poder finalizar el proyecto ha sido imprecindible el trabajo en equipo en temas de reparto de tareas, coordinación, etc.

Por otro lado, cabe recalcar que ha sido necesario una gran cantidad de tiempo para concluir el proyecto, puesto que, el tiempo utilizado supera al tiempo estimado para una asignatura de 4,5 créditos. Sin embargo, creo que el desarrollo de un proyecto grupal a elección ha sido positiva para el desarrollo de los alumnos en una orientación más profesional y, ha fomentado a la creatividad y al autoaprendizaje.

ÁLVARO: Desde mi punto de vista. Con la realización de este proyecto, hemos puesto en práctica todos los conocientos que hemos adquirido en el grado de electrónica. Hemos podido enfrentarnos a un proyecto real de ingenieria. Esto nos ha ayudado a adaptarnos a un ámbito mas profesional, donde podemos hacernos una idea de como es un equipo de ingenieria en la realidad. Esto me ha ayudado a desarrollar diferentes habilidades como: la capacidad de trabajar en un equipo grande, mayor habilidad para expresar mis ideas a mis compañeros, mayor capacidad de autoaprendizaje...

Además, el hecho de poder elegir el proyecto que ibamos a desarrollar me ha motivado mucho. Ya que he estado trabajando en cosas que realmente quería realizar. Por otra parte, cabe destacar que las horas de trabajo para lograr terminar el proyecto han sido bastante elevadas. A mi parecer, esta asignatura deberia ser de 6 créditos. Debido a la carga excesiva de trabajo que requiere, respecto a otras asignaturas de 4,5 creditos que he cursado a lo largo de la carrera.

JAIME: Desde mi perspectiva, la realización de este proyecto ha sido una oportunidad para poner en práctica todos los conocimientos adquiridos en el grado de electrónica. Nos hemos enfrentado a un proyecto real de ingeniería, lo que nos ha permitido adaptarnos a un entorno más profesional y obtener

una idea clara de cómo funciona un equipo de ingeniería en la realidad. Esta experiencia ha sido fundamental para el desarrollo de habilidades como trabajar en un equipo grande, expresar mis ideas de manera efectiva a mis compañeros y fortalecer mi capacidad de aprendizaje autónomo.

Además, el hecho de poder elegir el proyecto en el que trabajar nos ha motivado enormemente, ya que hemos tenido la oportunidad de enfocarnos en algo que realmente nos apasiona. No obstante, es importante mencionar que las horas de trabajo requeridas para completar el proyecto han sido considerablemente altas. En mi opinión, esta asignatura debería tener 6 créditos en lugar de los 4,5 créditos asignados, dado el nivel de carga de trabajo que exige en comparación con otras asignaturas que he cursado a lo largo de la carrera.

VALERIU: Tras abordar con éxito la finalización e implementación del proyecto, podemos concluir que este ha requerido de un buen equilibrio entre trabajo en equipo, cooperación y conocimientos técnicos. El enfrentarse a un proyecto real y complejo que involucra diversas disciplinas vistas a lo largo de la carrera ha dado pie a tener mayor conciencia de la importancia de tener una visión global de la Ingeniería, y en particular de la Ingeniería electrónica y sus diferentes ramas de diseño y cálculo. Se ha pretendido mostrar una aplicación práctica y útil de un sistema que involucra tanto Electrónica Analógica como Digital, trabajando en conjunto para lograr cubrir distintas funcionalidades útiles y propias de un hogar inteligente.

GONZALO: Por mi parte opino que el proyecto que hemos realizado es de una complejidad media, como se nos pedia en el desarrollo de la asignatura, nos hemos implicado mucho en su ejecucion porque creemos que lo merece. Por otro lado el resultado creo que podria ser algo mejor, aunque estoy muy contento con este, pero quizas de haber sabido la complejidad de algunos de los sistemas deberiamos habernos planteado el cambiarlos por otros algo mas sencillos. En general el trabajo en equipo ha sido impecable por parte de todos los integrantes. Finalmente me gustaria añadir que, aun que la asignatura es de 4.5 creditos, las horas usadas en el desarrollo de este proyecto y de la asignatura en general superan con creces a las esperadas o las impuestas por la guia de la asignatura. Aun con esto, esta asignatura deberia desarrollarse siempre de esta manera, dando lugar a que los alumnos desarrollen su creatividad y sus aptitudes mas potentes en el desarrollo de un proyecto que estos encuentren motivador y que englobe los conocimientos adquiridos a lo largo de todas las asignaturas de la carrera 😊