UNIVERSIDAD NACIONAL
DE LA PATAGONIA
SAN JUAN BOSCO

Ingeniería Electrónica

EE019 - Proyecto de Ingeniería Electrónica

Aplicaciones de Smart Building

Control de temperatura en ambientes calefaccionados por aire caliente

---

Alumnos:
Carabajal, Damián Nicolás
Delgado, Milton Damián
Tombesi, Federico

Tutor Académico:
Dr. Ramiro Peña

Profesores:
Dr. Berns, Daniel Walther
Dr. De Marziani, Carlos
Dr. Mayosky, Miguel Angel

Agradecimientos: Carabajal, Damián Nicolás

Dicen que parte del viaje es lo que sucede en el camino y, a decir verdad, tuve la fortuna de que la vida me haya sabido poner a las personas indicadas para poder llegar hasta aquí.

Antes que nada quiero agradecer a mi familia, a mis padres Ariela y Pablo, que con amor me han inculcado los valores de la vida, me han cuidado y han apoyado desde el principio. A mis hermanos Gastón y Valentín, que siempre tienen algo para charlar y me ayudaron a despejarme mas de una vez. A mis abuelos Blanca y Chabelo, que me guían donde estén y no sería la persona que soy si de ellos no hubiera aprendido sobre la paciencia, el cariño y el aprecio a las cosas simples de la vida. A mi primo Marcos, que me ha sabido escuchar, aconsejar y acompañar en toda la vida. A mi tía Iris, mi tío Mario y mi tío Enrique, que siempre supieron demostrarme su cariño y que me enseñaron que no hay que darse por vencido en la vida.

Un agradecimiento inmenso a mi compañera de vida Sol, que jamás me dejo bajar los brazos, que me escuchó y acompaño en cada momento, que me regalo risas, amor, comprensión y la felicidad de la vida.

A mis amigos y compañeros de proyecto Fede y Milton, que sin ellos este trabajo hubiera sido imposible de llevar adelante, estoy muy orgulloso de los profesionales en los que se convirtieron, pero más aún de las increíbles personas que son.

A mis amigos Nico y Fer, que me han acompañado en los días mas difíciles, me han regalado innumerable cantidad de risas, me han escuchado y hoy son mis hermanos de la vida.

A todos mis compañeros Nico "Mapache", Juani, Toto, Fede, Milton, Aarón, Franco, Fran, Gabi, Marcos, Julio, Lucas V., Lucas M. y todos los chicos del convivio que han hecho que los días sean mas llevaderos con risas y mates.

A todo el plantel docente que con su dedicación y pasión por la carrera han sabido despertar el interés en cada materia, en cada clase y en cada charla. A Javi, por estar siempre dispuesto a darnos una mano y a compartir unos mates. A Carlos, por el apoyo y guía para que este trabajo salga adelante. A Ramiro, que supo darnos las claves para poder resolver los inconvenientes que fueron presentando a lo largo del proyecto. A Juancito, por siempre tener la predisposición de ayudarnos y enseñarnos algo. Al profesor Pablo Carbone, por mostrarme esta carrera cuando tenia 15 años, que años mas tarde sin saberlo, nos volveríamos encontrar nuevamente en un aula.

A la universidad pública, que me dio la posibilidad de formarme, aprender y conocer a un grupo humano excelente en mi querida Universidad Nacional de la Patagonia San Juan Bosco, en especial al Departamento de Electrónica, que forma no solo ingenieros de calidad, sino que genera un excelente grupo humano año tras año.

"*Valentía es afrontar los temores para lograr lo que deseas sin dejar de ser uno mismo*"

Agradecimientos: Delgado, Milton Damián

Ha sido un largo camino el que tuve que recorrer para llegar hasta este punto, siento que quiero decir mil cosas y aún así me quedaría corto. Este proyecto marca el final de mi etapa como estudiante universitario, y sólo fue posible porque siempre estuve acompañado y parado sobre hombros de gigantes.

En primer lugar, quiero agradecer a mi familia, principalmente a mi padre Juan Luis Delgado Ojeda, y a mi hermana Yanina Vanesa Delgado, por su acompañamiento incondicional e infinita paciencia a lo largo de estos años, entendiendo mis tiempos y apoyándome en silencio.

A mis amigos de toda la vida, quienes no dejaron de invitarme a las juntadas y entendían mis ausencias o llegadas tarde, siempre tuvieron un lugar listo para mí cuando necesitaba despejarme y renovar energías.

A la universidad, especialmente al Departamento de Ingeniería Electrónica. Gracias por ser mucho más que una institución académica; gracias por ser un hogar. Tuve el privilegio de formarme entre profesores y alumnos que priorizan tanto el valor humano como el técnico. Agradezco a Susana y las demás cocineras, que trabajan con cariño para que tengamos un plato de comida todos los días. Por cosas así, defiendo con orgullo a la universidad pública que me permite estar hoy acá.

A mis profesores, por enseñarme mucho más que fórmulas y mostrarme que la ingeniería se construye también desde la empatía. Gracias por preocuparse genuinamente por nuestra formación, tanto académica como a nivel personal, gracias por esa calidez humana que crea un ambiente donde da gusto aprender.

A mis compañeros de carrera, que con el tiempo compartido se convirtieron en mis amigos. Gracias a Fede, Damián, Nico, Juan, Tomás, Aarón, Marcos, Gabi, Fran, Franco, Julio, Kevin, Emilio, Lucas V. y Lucas M. Sin ustedes, las materias hubieran sido cuesta arriba, muchas gracias por las incontables risas y el compañerismo brindado.

Especialmente quiero destacar a Aarón y Damián, quienes en más de una ocasión no permitieron que bajara los brazos y me animaron cuando el camino se ponía difícil. Si hoy estoy acá, es principalmente por ustedes.

A mis compañeros de proyecto, Damián y Fede. No soy capaz de visualizar este trabajo con nadie más que con ustedes. Gracias por demostrarme que las cargas se llevan mejor cuando son compartidas.

Por último, y más importante, dedico todo este esfuerzo a la memoria de mi madre, Fresia Uberlinda Vera Arenas. Siempre creíste en mí y me impulsaste a perseguir mis sueños. Aunque hoy no estés físicamente para ver este logro, espero desde el cielo puedas ver ésto y sentirte orgullosa. Te amo Má, todo lo que soy te lo debo a vos.

"*La única decisión posible es qué hacer con el tiempo que tenemos*"

Agradecimientos: Tombesi, Federico

Fue un largo viaje, el cual valió enormemente la pena transitar, pero no hubiese sido posible completarlo sin las personas que me rodearon todos estos años.

Agradezco a mi mamá Miryan, por el constante amor y su compañía durante toda mi vida al ayudarme a nunca bajar los brazos en nada, al darme la confianza necesaria para afrontar cualquier desafío, aplicado a lo académico y trascendiendo mucho más allá del mismo.

A mi papá Luis, por estar siempre y darme mucho cariño. A mi hermana Rosario por su compañía permanente y las charlas de la pasión por la ciencia y la tecnología, que me hacían permanecer enchufado siempre a este mundo que tanto me apasiona. También a mis hermanos Santiago, Pablo y Mariana.

A mis amigos de toda la vida Yomi, Juan Cruz, Fede, Branco y Mario, por ser un sostén emocional durante los años de carrera y ya hace más de 15 años (y contando...).

A todos los profesores del Departamento de Electrónica por la formación y el entusiasmo permanente en querer que uno aprenda, lo que siempre acompañó mi pasión por esta carrera y ayudó a materializar el entusiasmo en aprendizaje. A Carlos, por la predisposición a ayudarnos y el ánimo a seguir avanzando. A Ramiro, por ayudarnos a resolver inconvenientes que se fueron presentando. A Javi por acompañarnos algunos sábados para ayudarnos a abrir el laboratorio y compartir charlas y mates. También, destacar la calidez humana y la bondad que hay en el Departamento de Electrónica, es un lugar muy especial y que llevaré siempre en el corazón.

A la Universidad Pública, por la oportunidad y formación dada, pero también por darme grandes amigos como Milton y Damián, con quiénes tuve el placer de compartir muchos trabajos (siendo este el más especial) y todos los momentos lindos, son dos seres humanos espectaculares. No podría haber hecho este trabajo con nadie más que con ellos.

A Aarón que fue mi primer amigo de la Facultad, y lo es hasta el día de hoy.

A Daniel Roberto Fernández y Ramiro Ricardo Peña por haberme enseñado tanto en el Laboratorio de Automatización y Control, fue una gran experiencia poder aprender de tan buenos profesionales, son dos grandes profesores y sobretodo personas.

A Juancito, por las charlas permanentes en el pañol y el buen humor que muestra siempre.

A mis compañeros de carrera Nico, Tomi, Juan, Fran, Gabi, Marcos, Julio, Lucas V. y Lucas M., por las innumerables charlas y risas en el convivio.

A la cátedra de Física I por permitirme trabajar con ellos cómo auxiliar alumno, siempre fue un deseo involucrarme en la docencia y fue (y sigue siendo) una experiencia que me encanta y disfruto mucho.

Nada como la Universidad Pública

Resumen

El presente trabajo aborda el diseño e implementación de un sistema de control de temperatura en el marco de IoT (*Internet de las cosas*), con el objetivo de controlar la temperatura de la sala de reuniones del Departamento de Electrónica de la Universidad Nacional de la Patagonia San Juan Bosco (UNPSJB). Se busca garantizar confort térmico y maximizar la eficiencia de la distribución de calor del sistema de calefacción por aire existente.

Primero, se realizan modelos matemáticos para evaluar la dinámica de la temperatura en la sala de estudio y caracterizar de manera correcta el modelo de la planta térmica. Posteriormente, se calcula un controlador utilizando las metodologías aprendidas en las asignaturas de Control.

Luego, se diseña un difusor de caudal variable, y se fabrica mediante impresión 3D. Esto con el objetivo de reemplazar los difusores manuales actuales por el actuador personalizado controlado.

Finalmente, se integra una arquitectura de software para el monitoreo y control remoto del sistema y del actuador. Se implementa y ensaya en su totalidad, validando así su desempeño y contrastando los resultados con las simulaciones teóricas realizadas previamente.

Palabras clave: temperatura, controlador, sintonización, IoT, actuador/difusor, impresión 3D, MQTT.

Abstract

The present work addresses the design and implementation of a temperature control system within the framework of the Internet of Things (IoT), with the objective of regulating the temperature of the meeting room of the Department of Electronics at the Universidad Nacional de la Patagonia San Juan Bosco (UNPSJB). The aim is to ensure thermal comfort while maximizing the efficiency of heat distribution of the existing forced-air heating system.

First, mathematical models are developed to analyze the temperature dynamics of the studied room and to accurately characterize the thermal plant. Subsequently, a controller is designed using the methodologies acquired in control-related courses.

Next, a variable-flow air diffuser is designed and manufactured using 3D printing technology. This diffuser is intended to replace the existing manual diffusers with a customized, actively controlled actuator.

Finally, a software architecture for remote monitoring and control of the system and the actuator is integrated. The complete system is implemented and experimentally tested, thereby validating its performance and comparing the obtained results with the previously conducted theoretical simulations.

Keywords: temperature, controller, tuning, IoT, actuator/diffuser, 3D printing, MQTT.

Índice

Índice de figuras

1. Introducción

Mantener una temperatura adecuada en el lugar de trabajo es esencial para lograr un ambiente confortable y saludable. La Organización Mundial de la Salud (OMS) propone recomendaciones sobre la temperatura ideal en ambientes cerrados, según su informe publicado "*Housing and Health Guidelines*"[1] se sugiere mantener:

- Una temperatura mínima de 18°C en interiores para la mayoría de las personas saludables.
- Una temperatura de al menos 20°C para grupos vulnerables (como niños, personas mayores o con enfermedades crónicas).

Las recomendaciones se basan en evidencia científica sobre los efectos de frío en la salud, como **riesgos cardiovasculares y respiratorios**, se ha demostrado que temperaturas inapropiadas pueden afectar el sueño, el estado de ánimo y la salud mental. Si bien la OMS no define una temperatura máxima exacta para interiores en dicho informe, advierte el hecho de que elevadas temperaturas también son un riesgo para la salud.

En este contexto, tanto la OMS como otros organismos como la **Agencia de Protección Ambiental Europea (EEA)** o la **ASHRAE** (en EE.UU.) recomiendan mantener los interiores en un rango de confort térmico que no debe exceder los 26°C, y las razones son por riesgos de golpes de calor, mayor estrés cardiovascular, interferencia con el sueño y el descanso, posible deshidratación y fatiga.

El proyecto tiene como objetivo diseñar un control de temperatura modular para habitaciones que cuenten con un sistema de calefacción preexistente, consistente en un flujo de aire caliente que circula por conductos por encima del techo y cuyo ingreso o restricción de la entrada del mismo a las habitaciones se puede gobernar mediante la apertura o cierre de difusores.

El estudio se enfoca particularmente en el circuito de calefacción del Departamento de Ingeniería Electrónica (DIE), tomando como planta de análisis la Sala de Reuniones (Figuras 1a y 1b), o aula 345. Se busca entonces modelar la respuesta de esta habitación ante el ingreso del caudal de aire, con el fin de aplicar técnicas de control vistas a lo largo de la carrera para poder diseñar un controlador que cumpla con los requerimientos establecidos por el usuario.

Se busca que el sistema de control diseñado produzca que se cumpla con el nivel de confort térmico deseado por el usuario, es decir, la persona que vaya a ocupar la sala. Para ello se implementa un sensado de la temperatura de la habitación, cuyo valor se publica en un servidor para que el sistema de control pueda acceder a los datos, saber a qué temperatura se encuentra, y en función de la temperatura deseada que configure el usuario, mediante una aplicación o página web, el sistema decide la acción correspondiente sobre los difusores, con el objetivo de regular el paso del flujo de aire caliente hacia la sala.

(a) Puerta de la habitación

(b) Interior de la habitación

Figura 1: Habitación en estudio

En la Figura 2 se ve un esquema general de la idea del proyecto.

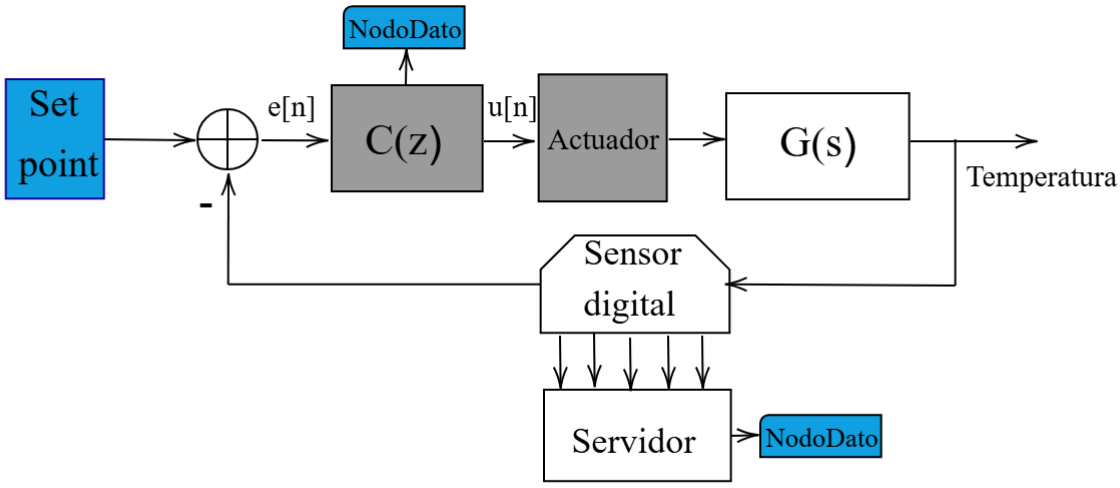


Figura 2: Diagrama de la idea general del proyecto

En donde:

- **Set point:** es la referencia de la temperatura que desea el usuario.
- **C(z):** controlador diseñado e implementado en un microcontrolador, encargado de calcular la acción de control a partir del error entre la referencia y la temperatura medida.
- **Actuador:** son los difusores mencionados anteriormente, los cuales en función de la señal de control $u[n]$ se abrirán o cerrarán en cierto grado, para dejar pasar o no el flujo de aire.
- **G(s):** es la planta en estudio, correspondiente a la dinámica de la sala. La cual se modela mediante una serie de mediciones.
- **Sensor digital:** dispositivo que se utiliza para medir la temperatura a la que se encuentra la habitación.
- **Servidor:** plataforma donde se suben los datos del sensor y el medio por el cual el controlador se informe de la temperatura a la cual se encuentra la habitación.

2. Marco teórico

2.1. Conceptos básicos en la transferencia de calor

Una de las formas de energía que aparecen con frecuencia en la naturaleza es el calor, el cual se interpreta como energía que se puede transferir de un sistema a otro, siendo siempre de un medio de mayor temperatura a otro de menor temperatura, este proceso termina cuando los medios conciben la misma temperatura [2].

El calor se puede transferir a través de tres modos:

- **Conducción**: es el proceso de transferencia de energía a través del contacto directo sin intercambiar materia.
- **Convección**: es el proceso de transferencia de energía entre un sólido y un líquido o gas que se encuentra en movimiento.
- **Radiación**: es el proceso de intercambio de energía emitida por materia en forma de ondas electromagnéticas.

El **flujo de calor** es un flujo de energía por unidad de área por unidad de tiempo, que comúnmente se denota como $\vec{\phi_q}$.

El flujo se describe adecuadamente por la Ley de Fourier que se representa por:

$$\vec{\phi_q} = -k\nabla T \tag{1}$$

donde σ es la conductividad térmica, mientraas que el signo representa que el flujo de calor se desplaza desde la zona de temperatura más alta a la zona con temperatura más baja.

La **conductividad térmica** es una propiedad de la materia que indica la tasa de transmisión de calor del material [3], por unidad de área y por unidad de diferencia de temperatura $\left[\frac{W}{m^2 K}\right]$. Esta propiedad depende de la densidad, porosidad y humedad del material.

La **difusividad térmica** influye igual que la conductividad térmica, con la diferencia de que la consuctividad térmica se manifiesta en estados estacionarios y la difusividad se presenta en los estados transitorios. La difusividad térmica dpende de tres propiedades físicas, la conductividad térmica, el calor específico y la densidad del material:

$$\sigma = \frac{k}{\rho\, C_p}\ (m/s^2) \tag{2}$$

2.2. Sistemas térmicos

Los sistemas térmicos son aquellos que involucran la transferencia de calor de una sustancia a otra [4]. Estos sistemas se analizan en términos de resistencia y capacitancia, aunque la capacitancia térmica y la resistencia térmica tal vez no se representen

con precisión como elementos de parámetros concentrados ya que, por lo general, están distribuidos en todas las sustancias. Para lograr análisis precisos, deben utilizarse modelos de parámetros distribuidos. Pero, pero simplificar el análisis, se considerará en el desarrollo del trabajo, que un sistema térmica se representa mediante un modelo de parámetros concentrados, que las sustancias que se caracterizan por una resistencia al flujo de calor tienen una capacitancia térmica insignificante y que las sustancias que se caracterizan por una capacitancia térmica tienen una resistencia insignificante al flujo de calor.

El calor fluye de una sustancia a otra de las tres formas ya descriptas. La transferencia de calor por radiación sólo se aprecia si la temperatura del emisor es muy alta en comparación con la del receptor [2]. La mayor parte de los procesos térmicos en los sistemas de control de procesos no involucran transferencia de calor por radiación. Para la transferencia por conducción o convección:

$$q = K\ \Delta\theta \tag{3}$$

donde:

- q= flujo de calor, $kcal/seg$;
- $\Delta\theta$ = diferencia de temperatura, °C;
- K= coeficiente, $kcal/seg \circ C$.

el coeficiente K se obtiene mediante:

$$\begin{aligned} K &= \frac{kA}{\Delta X} \quad \text{por conducción} \\ &= HA \quad \text{por convección} \end{aligned}$$

donde:

- k=conductividad térmica, $kcal/m\,seg\ \circ C$;
- A= área normal para flujo de calor, m^2;
- ΔX = espesor del conductor, m;
- H= coeficiente de convección, $kcal/m^2\,seg \circ C$.

Resistencia y capacitancia térmicas: La resistencia térmica R para la transferencia de calor entre dos sustancias se define del modo [4]:

$$R = \frac{cambio\ en\ la\ diferencia\ de\ temperatura\ ,\ \circ C}{cambio\ en\ el\ flujo\ de\ calor\ ,\ kcal/seg}$$

Para una transferencia por conducción o convección, se obtiene la resistencia térmica mediante:

$$R = \frac{d(\Delta\theta)}{dq} = \frac{1}{K} \tag{4}$$

como los coeficientes de conductividad y convección térmica con casi constantes, laa resistencia térmica para la conducción o la convección es constante.

La capacitancia térmica C se define mediante:

$$R = \frac{cambio\ en\ el\ calor\ almacenado\ ,\ [kcal]}{cambio\ en\ la\ temperatura\ ,\ [^{\circ}C]}$$

o bien:

$$C = m\,c \tag{5}$$

donde:

- m= masa de la sustancia considerada, $[kg]$.
- c= calor específico de la sustancia, $[kcal/kg^{\circ}C]$.

2.3. Medición de temperatura

La temperatura es la magnitud física que caracteriza el movimiento aleatorio medio de las moléculas de un cuerpo físico. Es representativo del estado termodinámico de un cuerpo, y su valor está determinado por la dirección del flujo neto de calor entre dos cuerpos.

La temperatura, expresada en Kelvin, es la temperatura básica. También es muy usada la temperatura en grados Celsius, definida por:

$$T(^{\circ}C) = T(K) - 273,15 \tag{6}$$

2.4. Identificación de sistemas

La identificación de sistemas trata con el problema de construir modelos matemáticos de sistemas dinámicos, basados en los datos observados de los mismos. Es un proceso alternativo para obtener un modelo, cuando no es posible encontrar un conjunto de ecuaciones diferenciales que describan el comportamiento dinámico del sistema [5].

La identificación del modelo del sistema basada en datos reales involucra tres pasos básicos:

- **Registro de datos:** Se define un experimento, en el cual se discriminan las señales que serán medidas y que tipos de entradas son apropiadas para aplicar al proceso.
- **Definición de modelos propuestos:** En este paso, se definen una serie de modelos candidatos. Existen dos casos básicos:
  1. Identificación como caja gris, sucede cuando hay algo de información acerca de los modelos candidatos basados en la física del proceso, siendo el objetivo encontrar los parámetros del modelo mediante la interpretación física.

2. Identificación como caja negra, cuando no hay información acerca del proceso físico en estudio y los parámetros usados son ajustados desde el modelo propuesto según los datos observados.

- **Determinación del mejor modelo, basada en los datos disponibles:** Este paso incluye una verificación de la calidad del modelo, que consiste en chequear como se comportan los modelos en comparación a los datos reales.

2.4.1. Método basado en la respuesta al escalón:

La prueba más usada para identificar el modelo de un proceso es la de la respuesta al escalón, que permite obtener modelos de primer orden (FOPDT - *First Order Plus Dead Time*) y de segundo orden (SOPDT- *Second Order Plus Dead Time*), que son normalmente usados para la sintonización de controladores PID [5].

Los métodos más simples son los métodos gráficos, y permiten obtener los parámetros de una función de transferencia. Considerando un modelo FOPDT:

$$G(s) = \frac{K_p}{1 + T\,s}\, e^{-L\,s} \tag{7}$$

Con ello, ya se hacen los dos primeros pasos propios de una identificación de sistemas (estructura del modelo y definición de las pruebas a realizar). El tercer paso consiste en determinar K_p, T y L que mejor se ajusten a los datos. En la Figura 3 se representa la salida de una planta y la acción de control por un escalón de amplitud $U_F - U_I$ aplicado en $t = t_0$ cerca de un punto de operación inicial (U_I,Y_I).

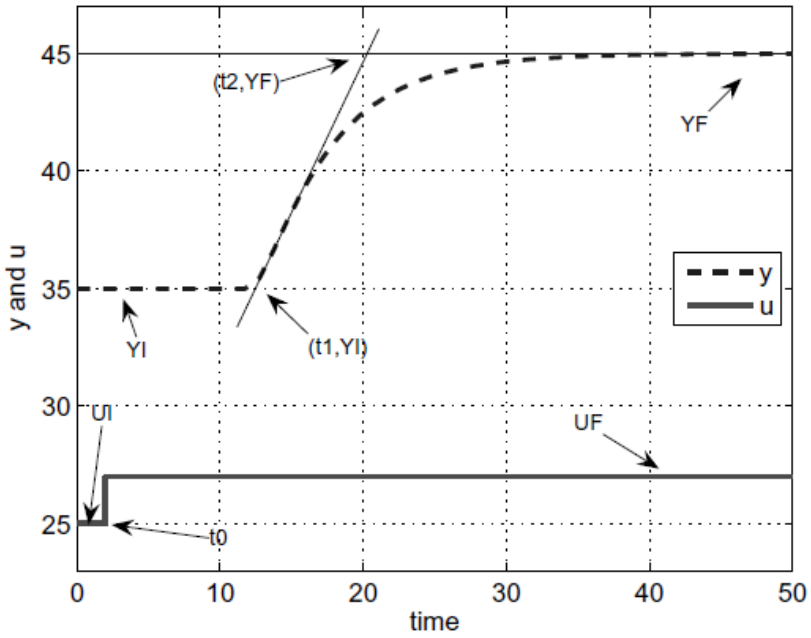


Figura 3: Identificación de K_p, L y T en una prueba al escalón

La salida va desde un estado estático inicial Y_I hasta el estado estático final Y_F.

Se dibuja una tangente en el punto en el cual posea la máxima pendiente y se obtienen los puntos marcados en la Figura 3. Con esa información, se calculan:

$$L = t_1 - t_0 \tag{8}$$

$$T = t_2 - t_1 \tag{9}$$

$$K_p = \frac{Y_F - Y_I}{U_F - U_I} = \frac{\Delta Y}{\Delta U} \tag{10}$$

Los pasos para el proceso de identificación son:

1. **Paso 1:** Seleccionar una entrada escalón ΔU.
2. **Paso 2:** Aplicar el escalón en un t_0 y registrar los valores de salida de la planta hasta que el sistema alcance la condición de estado estacionario de vuelta.
3. **Paso 3:** Determinar ΔY, t_1 y t_2
4. **Paso 4:** Obtener K_p, T y L usando las Ecuaciones 8, 9 y 10.

2.5. Controladores de temperatura

Un controlador de temperatura es un instrumento usado para controlar la temperatura de un sistema físico. El controlador de temperatura tiene entradas procedentes de un **sensor de temperatura** una temperatura de referencia, y posee una salida que está conectada a un elemento de control tal como un calentador o ventilador.

Para regular con precisión la temperatura del proceso sin la participación continua del operador, un sistema de control de temperatura se basa en un regulador, el cual acepta un sensor de temperatura tal como un termopar o RTD como entrada. Se compara la temperatura real a la temperatura de control deseada (referencia), o punto de ajuste, y proporciona una salida a un elemento de control. El regulador de temperatura solo es una parte del sistema de control, y todo el sistema debe ser analizado para elegir un controlador adecuado.

Los tipos diferentes de control de temperatura son:

- Controlador de temperatura On / Off: es la forma más simple de control de temperatura. La salida del regulador está encendida o apagada, sin un estado medio. Un controlador de temperatura ON/OFF cambia la salida sólo cuando la temperatura atraviesa el punto de ajuste. Para el calentamiento, la salida se activa cuando la temperatura está por debajo del punto de ajuste, y se apaga cuando está por encima del mismo. Cada vez que la temperatura cruza el punto de ajuste, el estado de la salida cambia, la temperatura del proceso oscila continuamente, entre el punto de ajuste. En los casos en que este ciclo se produce rápidamente, y para evitar daños a los contactores y válvulas, se añade un diferencial de encendido y apagado, o "histéresis", a las operaciones del controlador de temperatura. Este diferencial requiere que la temperatura exceda del punto de ajuste por una cierta cantidad antes de que se active o desactive de nuevo.

- Controlador de temperatura proporcional: este control elimina el ciclo asociado del control ON-OFF. Un controlador proporcional disminuye la potencia media suministrada al calentador cuando la temperatura se aproxima al punto de ajuste. Esto tiene el efecto de disminuir la energía del calentador al aproximarse al punto de ajuste sin que lo sobrepase, mantenimiento una temperatura estable. Esta dosificación se puede realizar girando el encendido y apagado de salida para intervalos cortos de tiempo. La "proporcionalización de tiempo"varía la relación de tiempo **ON** y tiempo **OFF** para controlar la temperatura. La acción proporcional se produce dentro de una "banda proporcional en torno a la temperatura objetivo. Fuera de esta banda, el controlador de temperatura se comporta como una unidad ON/OFF normal, con la salida, ya sea totalmente ON (por debajo de la banda) o totalmente OFF (por encima de la banda).
- Controladores PID: es el tercer tipo de control de temperatura, el cual ofrece una combinación del proporcional con control integral y derivativo. De hecho las siglas PID hacen referencia a un control Proporcional Integral Derivativo. El regulador proporciona es el control más preciso y estable de los tres tipos de controladores, y se utiliza comúnmente en sistemas que tienen una masa relativamente pequeña, que son aquellos que reaccionan rápidamente a cambios en la energía añadida al proceso.

  Se recomienda en sistemas en los que la carga cambia a menudo y no se espera que el controlador lo compense automáticamente, debido a los frecuentes cambios en el punto de referencia, la cantidad de energía disponible, o la masa a controlar. El controlador de temperatura PID combina el control proporcional con dos ajustes adicionales, que ayuda a la unidad automáticamente a compensar los cambios en el sistema.

2.6. Controladores PID

Los controladores PID son ampliamente usados en todo tipo de industrias. En control de procesos, mas del 95 % de los lazos de control son PID, debido al hecho de que los PID permiten, con un simple algoritmo de realimentación, eliminar errores de estado estacionario y lograr un rendimiento razonable a lazo cerrado [6].

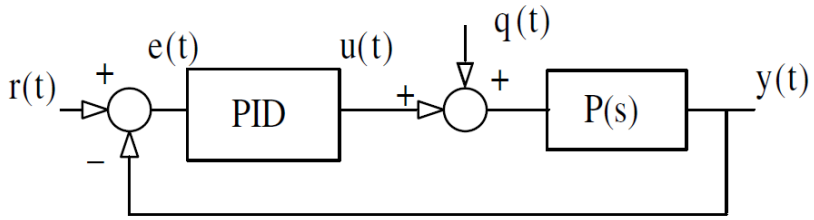


Figura 4: Sistema de control con un lazo PID

La acción de control de un controlador PID ideal $u(t)$ puede ser expresada como:

$$u(t) = K_c \left[e(t) + \frac{1}{T_i} \int_0^t e(\tau)\, d\tau \; + \; T_d \frac{de(t)}{dt} \right] \tag{11}$$

donde $e(t) = r(t) - y(t)$ es el error entre las señales de referencia $r(t)$ y la de salida $(y(t)$ (ver Figura 4). K_c es la ganancia proporcional, T_i es el tiempo integral y T_d es el tiempo derivativo.

La función de transferencia de este controlador es:

$$C(s) = \frac{U(s)}{E(s)} = K_c \left(1 + \frac{1}{T_i s} + T_d s \right) \tag{12}$$

2.6.1. Sintonización Ziegler-Nichols

El método de sintonización más famoso para controladores PID, es la regla de Ziegler-Nichols, que fue desarrollada usando simulaciones con diferentes sistemas donde la relación entre el tiempo muerto L y la constante de tiempo T satisface $\frac{L}{T} < 1$ [6].

Hay dos métodos de Ziegler-Nichols para la sintonización de PID's en la forma estándar. La primera, conocido como método Ziegler-Nichols a lazo abierto o curva de reacción, que es la que se utiliza y asume que la respuesta al escalón a lazo abierto de la planta puede ser modelada por:

$$P(s) = \frac{a\, e^{-Ls}}{s\, T} = \frac{a\, e^{-\tau_o s}}{s\, v_o} \tag{13}$$

los parámetros para la forma estándar del PID son dados en la Fig.5 donde $K_o = 1$.

	$\mathbf{K_p}$	$\mathbf{T_r}$	$\mathbf{T_d}$
P	$\frac{\nu_o}{K_o \tau_o}$		
PI	$\frac{0.9\nu_o}{K_o \tau_o}$	$3\tau_o$	
PID	$\frac{1.2\nu_o}{K_o \tau_o}$	$2\tau_o$	$0.5\tau_o$

Figura 5: Reglas de sintonización Z-N

2.6.2. Sintonización Cohen-Coon

Otro método de sintonización bien conocido es el de Cohen-Coon. Este método es basado en un modelo FOPDT y usa un proceso de ubicación de polos dominantes. Para la forma estándar de los PID, los parámetros son presentados en la Figura 6 [6]:

	$\mathbf{K_p}$	$\mathbf{T_r}$	$\mathbf{T_d}$
P	$\frac{\nu_o}{K_o\tau_o}\left[1+\frac{\tau_o}{3\nu_o}\right]$		
PI	$\frac{\nu_o}{K_o\tau_o}\left[0.9+\frac{\tau_o}{12\nu_o}\right]$	$\frac{\tau_o[30\nu_o+3\tau_o]}{9\nu_o+20\tau_o}$	
PID	$\frac{\nu_o}{K_o\tau_o}\left[\frac{4}{3}+\frac{\tau_o}{4\nu_o}\right]$	$\frac{\tau_o[32\nu_o+6\tau_o]}{13\nu_o+8\tau_o}$	$\frac{4\tau_o\nu_o}{11\nu_o+2\tau_o}$

Figura 6: Reglas de sintonización Cohen-Coon

Este método permite una mejor sintonización para sistemas que tengan una constante de tiempo elevada, sistemas con más inercia.

2.7. IoT - Protocolo MQTT

El Internet de las cosas (IoT), es un concepto que se define como la interacción digital de objetos cotidianos en internet, orientado a conectar dispositivos (cosas) antes que personas. A la hora de planear una aplicación IoT [7], se puede usar el protocolo de servicios web HTTP, de manera que el dispositivo pueda enviar datos como un pedido y recibirlos como una respuesta HTTP.

Este modo de transmisión de datos posee algunas limitaciones:

- **Protocolo sincrónico:**En un protocolo sincrónico, el cliente debe esperar a que el servidor responda antes de continuar. Esto funciona bien en los navegadores web tradicionales, donde la comunicación suele ser estable y con baja latencia.

  Sin embargo, en entornos IoT —con muchos dispositivos conectados, redes inalámbricas, posibles pérdidas de paquetes o latencias elevadas— este modelo puede generar problemas, ya que obliga al dispositivo a quedarse "bloqueado" esperando la respuesta del servidor. Esto consume tiempo, energía y recursos. Por esta razón, los protocolos sincrónicos no suelen ser la mejor opción en IoT. En cambio, los protocolos asincrónicos resultan más adecuados porque permiten que el dispositivo envíe los datos y continúe con su funcionamiento sin esperar una respuesta inmediata, dejando que la red o el servidor procesen la información cuando sea posible.

- **Protocolo de una sola dirección:** el cliente debe iniciar la conexión. En una aplicación IoT, el dispositivo suele ser un cliente y significa que no pueden recibir comandos por la red.
- **Protocolo de uno a uno:** el cliente realiza un pedido, y el servidor responde. Es dificil y caro realizar un mensaje de difusión a todos los dispositivos de la red.
- **Pesado:** con gran cantidad de cabeceras y reglas, por lo que no es recomendable para redes limitadas.

El MQTT (*Message Queuins Telemetry Transport* - Transporte de mensajes de cola de telemetría) es un protocolo de transporte de mensajes que utiliza el modelo cliente-servidor, y se desarrolla en 1999 por IBM. Es liviano y flexible, de código abierto, simple y diseñado para ser fácil de implementar, resultando conveniente para comunicación máquina a máquina (M2M). Entre sus principales características, se destacan:

- Poporciona calidad de servicio configurable en los datos transmitidos;
- Eficiente en ancho de banda y consumo de potencia;
- Se abstrae de los datos que envía;
- Detecta la sesión persistente.

2.7.1. Modelo de publicar/suscribirse

El modelo *publicar/suscribirse* es una alternativa al modelo tradicional de cliente-servidor, y permite desacoplar el emisor, como puede ser un cliente que publica un mensaje, del suscriptor, que es el destinatario final del mensaje. Esto significa que tanto el emisor como el receptor desconocen la existencia del otro. El nexo entre ambos es el servidor, osea el **broker**, que es conocido por lo clientes involucrados (ver Figura 7). El "broker" se encarga de filtrar los mensajes entrantes y los distribuye a los destinatarios según el tema del mensaje (cada emisor debe indicar el tema del mensaje publicado que le interesa) [8].

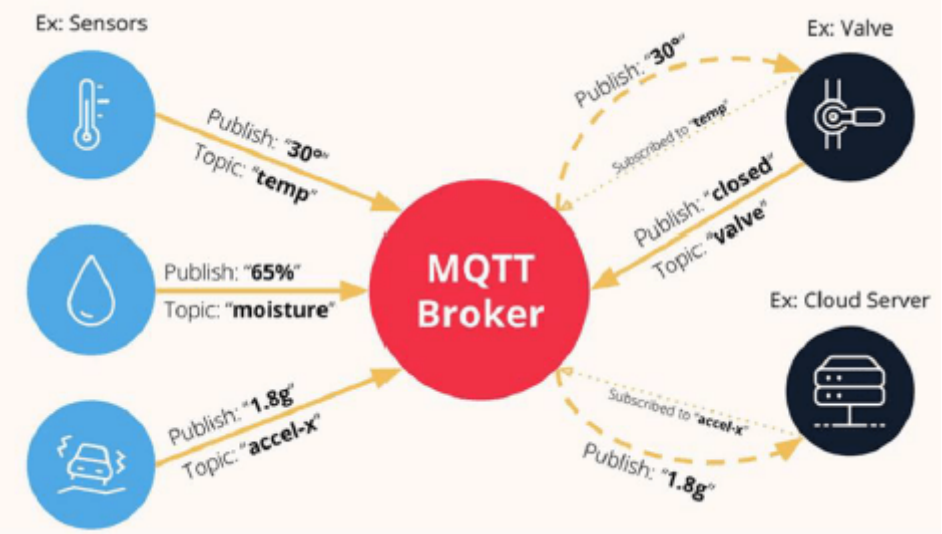


Figura 7: Modelo de publicar/suscribirse

La principal característica de este modelo, es el **desacople** del emisor y el suscriptor, que puede identificarse como:

- **Desacople espacial:** tanto el emisor como el receptor se desconocen.
- **Desacople temporal:** el emisor y el receptor no deben estar sincronizados.

Otra característica importante es la escalabilidad que presenta este modelo, que implica que el agregar un cliente sea simple. El broker puede guardar mensajes destinados a clientes fuera de línea y, en general, la comunicación es asincrónica, así no se detienen otras tareas de procesamiento.

2.7.2. Conceptos de MQTT

- **Cliente:** el cliente se define como cualquier dispositivo que entiende el protocolo MQTT y se conecta a un broker MQTT por medio de cualquier tipo de red. Desde un microcontrolador hasta un servidor.
- **Broker:** es el responsable de recibir todos los mensajes emitidos, filtrarlos, decidir a quién se les retransmiten y retransmitirlos a todos los clientes suscritos. Aparte de ello, también debe encargarse de mantener la sesión abierta de todos los clientes persistentes, autenticar y autorizar los clientes.
- **Conexión MQTT:** este protocolo está basado en el stack TCP/IP, y corresponde en el modelo OSI a las capas superiores a la de transporte. En la Figura 8 se observa la conexión que se establece siempre entre un cliente y el broker. Se inicia con el cliente enviando un mensaje de control al broker, y este responde con un paquete de acuse de recibo y un código de estado. Una vez establecida la conexión, como es persistente, el broker debe mantenerla abierta hasta que el cliente emita un comando de desconexión, o que directamente se pierda la conexión.

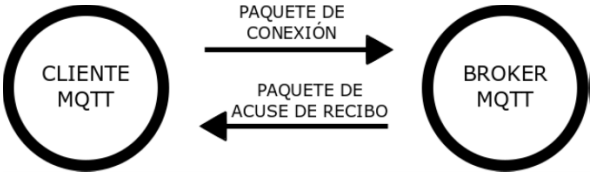


Figura 8: Representación del envío de paquetes MQTT

- **Publicar:** cuando un cliente se conecta a un broker, puede publicar mensajes con determinada calidad de servicio. Como el protocolo se abstrae de los datos enviados, pueden transmitirse datos binarios, texto o incluso XM.
- **Suscribirse:** para suscribirse a un tema, un cliente debe enviar un paquete de suscripción, el cual acusa recibo de cada pedido y la suscripción es exitosa.
- **Calidad de servicio:** MQTT puede funcionar sobre redes no confiables, por lo que prevé tres formas de calidad de servicio (QoS), y permite al cliente especificar la confiabilidad deseada [8].
  - **QoS 0:** no requiere que el cliente acuse recibo, y la confiabilidad va a ser la misma que la de capa de red (TCP/IP).
  - **QoS 1:** asegura que el mensaje llegue una vez, pero pueden ocurrir duplicados, por lo tanto el cliente debe acusar recibo del paquete. El receptor puede manejar el caso depaquetes duplicados.
  - **QoS 2:** asegura que el mensaje llegue una vez y solo una vez. Requiere un handshake de cuatro paquetes, por lo tanto disminuye la performance del broker. Además, requiere mayor procesamiento por parte del cliente.
- **Temas:** el tema consiste en una cadena codificada en UTF-8 (formato de codificación de caracteres), la cual usa el broker para filtrar los mensajes a cada cliente. Un tema consiste en uno o varios niveles de tema, y presenta una estructura jerárquica.

2.8. Smart Buildings

Los Smart Buildings son aquellos cuyas instalaciones y sistemas (de climatización, iluminación, electricidad, seguridad, telecomunicaciones, multimedia, informáticas, control de acceso, etc.) permiten una gestión y control integrada y automatizada, con el fin de aumentar la eficiencia energética, la seguridad, la usabilidad y la accesibilidad.

El concepto de Smart Buildings es aplicable para todas las tipologías de ambientes, tanto para su rehabilitación como para la nueva construcción . Oficinas, hospitales, hoteles, bancos, museos, casas, etc, todos son susceptibles de convertirse en ambientes inteligentes.

Los objetivos de un Smart Building son los siguientes:

- Satisfacer las necesidades presentes y futuras de los ocupantes, propietarios y operadores del edificio.
- La flexibilidad, tanto en la estructura como en los sistemas y servicios.
- La funcionalidad del edificio.
- Mayor confort para el usuario.

Y los objetivos desde un punto de vista tecnológico son:

- La disponibilidad de medios técnicos avanzados de telecomunicaciones.
- La automatización de las instalaciones.
- La integración de servicios.

3. Interpretación matemática del problema

La habitación en estudio se encuentra bajo la influencia de dos factores de interés: los cambios de temperatura en el exterior de la misma y el sistema de calefacción central del edificio; con este último, mediante un termostato, se elige la temperatura deseada. Una vez alcanzada, el sistema se desactiva automáticamente.

Pasado un tiempo, la temperatura de la habitación varía a causa de la temperatura exterior, activándose nuevamente el sistema de calefacción y generando un proceso cíclico.

Una explicación más detallada de todo este proceso se encuentra a lo largo de la Sección 4: *Circuito de calefacción pre-existente*, donde se explican los distintos componentes del sistema de calefacción de la universidad.

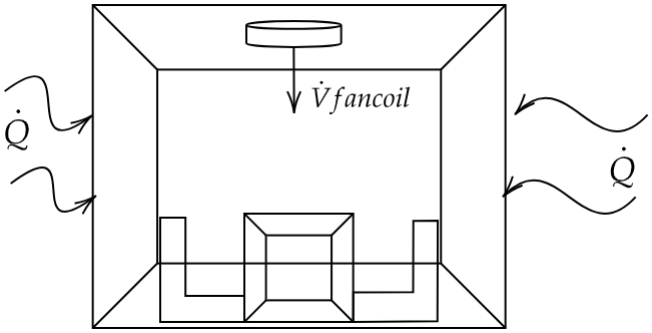


Figura 9: Esquema de la habitación

Se analiza la transferencia de calor en una de las paredes de la habitación bajo estudio. Se tienen las siguientes consideraciones:

- la pared es homogénea.
- su estructura no presenta irregularidades en las uniones del material que la compone.
- desaparecen las condiciones naturales de contornos, tanto de flujo calórico como de temperatura de cada región que constituye la pared.
- el objeto de estudio será una celda, interpretada como una región simple y repetitiva de toda la estructura.

3.1. Matemática del problema

La celda de la Figura 10 representa una región en el espacio (x, y, z): $\Omega := (0, a)\, x\, (-\frac{b}{2}, \frac{b}{2})\, x\, (-\frac{c}{2}, \frac{c}{2})$, donde a es la profundidad del ladrillo, b representa el largo y c la altura.

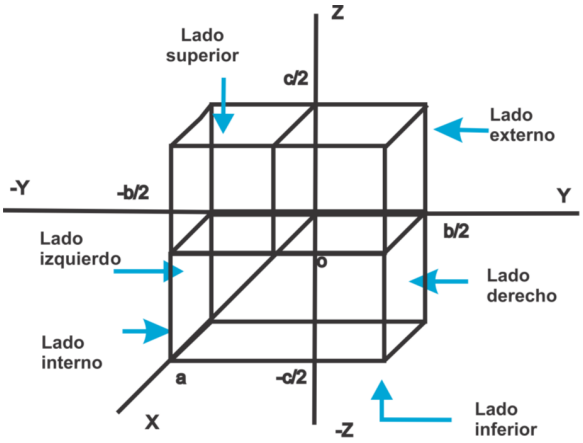


Figura 10: Celda mínima de la habitación

En cada uno de los lados de la celda Ω se definen condiciones de frontera referentes al evento físico, en el caso de los lados correspondientes al eje x se definen condiciones de tipo Dirichlet, que son: una temperatura constante T_{int}, que es la que se desea mantener, para el lado interior y $f(t)$, que describe la temperatura a lo largo del día, para el lado exterior.

Por la forma en que se ordenan las celdas a lo largo de la estructura de una pared, y la repetición continua de ellas, las condiciones que se definen para los lados derecho e izquierdo correspondientes al eje y son de tipo periódicas, al igual que para las condiciones correspondientes a la dirección del eje z, resumidas en la siguiente tabla [9]:

	Descripción	Condiciones de frontera
$L_{int} := \{(a, y, z) : -\frac{b}{2} \leq y \leq \frac{b}{2}, -\frac{c}{2} \leq z \leq \frac{c}{2}\}$	Lado interno	Dirichlet
$L_{ext} := \{(0, y, z) : -\frac{b}{2} \leq y \leq \frac{b}{2}, -\frac{c}{2} \leq z \leq \frac{c}{2}\}$	Lado externo	Dirichlet
$L_{der} := \{(x, \frac{b}{2}, z) : 0 \leq x \leq a, -\frac{c}{2} \leq z \leq \frac{c}{2}\}$	Lado derecho	Periódicas
$L_{izq} := \{(x, -\frac{b}{2}, z) : 0 \leq x \leq a, -\frac{c}{2} \leq z \leq \frac{c}{2}\}$	Lado izquierdo	Periódicas
$L_{sup} := \{(x, y, \frac{c}{2}) : 0 \leq x \leq a, -\frac{b}{2} \leq y \leq \frac{b}{2}\}$	Lado superior	Periódicas
$L_{inf} := \{(x, y, -\frac{c}{2}) : 0 \leq x \leq a, -\frac{b}{2} \leq y \leq \frac{b}{2}\}$	Lado inferior	Periódicas

Al considerar las ecuaciones de Fourier para el estudio de la transferencia de calor y las condiciones descritas [9]:

$$\frac{\partial \overline{T}(x,y,z,t)}{\partial t} = \sigma \left(\frac{\partial^2 \overline{T}(x,y,z,t)}{\partial x^2} + \frac{\partial^2 \overline{T}(x,y,z,t)}{\partial y^2} + \frac{\partial^2 \overline{T}(x,y,z,t)}{\partial z^2} \right), \qquad (14)$$

Transferencia de calor en 3 dimensiones

$$\overline{T}(a,y,z,t) = T_{int},\ \ \overline{T}(0,y,z,t) = f(t),\ \ -\frac{b}{2} \leq y \leq \frac{b}{2},\ \ -\frac{c}{2} \leq z \leq \frac{c}{2},\ \ t \geq 0, \tag{15}$$

Condiciones de Dirichlet.

$$\overline{T}(x,\frac{b}{2},z,t) = \overline{T}(x,-\frac{b}{2},z,t),\ \ \frac{\partial \overline{T}(x,\frac{b}{2},z,t)}{\partial y} = \frac{\partial \overline{T}(x,-\frac{b}{2},z,t)}{\partial y},\ \ 0 \leq x \leq a,\ \ -\frac{c}{2} \leq z \leq \frac{c}{2}; \tag{16}$$

Condiciones de frontera periódicas en y

$$\overline{T}(x,y,\frac{c}{2},t) = \overline{T}(x,y,-\frac{c}{2},t),\ \ \frac{\partial \overline{T}(x,y,\frac{c}{2},t)}{\partial z} = \frac{\partial \overline{T}(x,y,-\frac{c}{2},t)}{\partial z},\ \ 0 \leq x \leq a,\ \ -\frac{b}{2} \leq y \leq \frac{b}{2}; \tag{17}$$

Condiciones de frontera periódicas en z

$$\overline{T}(x,y,z,0) = \overline{T}_0(x,y,z), \tag{18}$$

Condición inicial

donde $f(t)$ es una función que describe la variación de la temperatura externa en un día.

Se considera que las temperaturas en las caras exterior e interior de la pared no dependen de las variables espaciales y y z, por lo que $T_0(x,y,z)$ de la pared tiene que depender solo de la variable x y hay que hallar su expresión. $T_0(x,y,z)$ denota la distribución estacionaria de la temperatura en la pared y por ello se supone que satisface la ecuación de calor estacionaria:

$$\Delta T_0 = \frac{\partial^2 T_0}{\partial x^2} + \frac{\partial^2 T_0}{\partial y^2} + \frac{\partial^2 T_0}{\partial z^2} = 0 \tag{19}$$

y las condiciones de contorno:

$$T_0(a,y,z) = T_{int},\quad T_0(0,y,z) = f(0) \tag{20}$$

Este problema tiene solución única y, por lo tanto, si se halla una solución que dependa únicamente de la variable x ésta tiene que ser solución única [10].

Pero, si T_0 dependiera solamente de la variable x, se tendría:

$$\frac{d^2 T_0(x)}{dx^2} = 0, \tag{21}$$

$$T_0(a) = T_{int},\quad T_0(0) = f(0). \tag{22}$$

De 21, se tiene que:

$$T_0(x) = \alpha\, x + \beta \tag{23}$$

y de 22 se obtiene $\beta = f(0)$ y $T_0(a) = \alpha\, a + f(0) = T_{int}$, de donde $\alpha = \dfrac{T_{int} - f(0)}{a}$, osea que $T_0(x)$ debe ser de la forma:

$$T_0(x) = (T_{int} - f(0))\,\frac{x}{a} + f(0), \tag{24}$$

Pero, una vez se sabe que $T_0(x)$ tiene la forma 24, entonces por la unicidad de la solución del sistema (14) a (18) se concluye que si ese sistema tiene alguna solución que dependa de la variable x, ésta tiene que ser una única solución [10].

La solución del sistema (14) a (18) que depende únicamente de la variable x satisface el sistema [10] [11]:

$$\frac{\partial \overline{T}}{\partial t} = \sigma\,\frac{\partial^2 \overline{T}}{\partial x^2}, \qquad 0 < x < a, \tag{25}$$

$$\overline{T}(0,t) = f(t), \qquad \overline{T}(a,t) = T_{int} \tag{26}$$

$$\overline{T}(x,0) = T_0 = (T_{int} - f(0))\,\frac{x}{a} + f(0), \tag{27}$$

Las condiciones de periodicidad (16) y (17) desaparecen cuando se supone que $\overline{T}$ no depende ni de y ni de z. Pero, se sabe que el sistema (25)-(27) tiene solución única que es también solución única del sistema (14) a (18), la cual, no depende ni de y ni de z.

3.2. Solución analítica del modelo presentado

El problema planteado es un problema auxiliar del problema de la **Ecuación de calor con condiciones de frontera de Dirichlet no homogéneas** [12], con la diferencia de que la condición de contorno depende del tiempo, para $\overline{T}(0,t) = f(t)$.

Si se propone el cambio de variable [12]:

$$\overline{T}(x,y,z,t) = \overline{T}(x,y,z,t) - H_0(x,t), \tag{28}$$

donde

$$H_0(x,t) = (T_{int} - f(t))\,\frac{x}{a} + f(t), \tag{29}$$

entonces el problema (14)-(18) se convierte en un problema con condiciones de contorno homogéneas:

$$\frac{\partial T}{\partial t} = \sigma\,\frac{\partial^2 T}{\partial x^2} + \left(\frac{x}{a} - 1\right)\frac{df}{dt}, \tag{30}$$

$$T(0,t) = T(a,t) = 0 \tag{31}$$

$$T(x,0) = 0 \tag{32}$$

Al hacer el cambio de variable de $\overline{T}$ a T, la ecuación diferencial 30 que se obtiene para T no es homogénea.

Es un resultado conocido que la ecuación de calor con condiciones de frontera de Dirichlet no homogéneas (la cual al aplicar el cambio de variable queda con condiciones homogéneas). Se pueda expresar en la forma [12][13]:

$$s(x,t) = \sum_{n=1}^{\infty} B_n \, e^{-\sigma\,(n\,\pi/a)^2\,t} \, sen\left(\frac{n\pi x}{a}\right), \tag{33}$$

siendo esta quien sugiere una buena base ortogonal en $L_2(0,a)$ para expresar la solución del problema (30)-(32) es precisamente $sen(\frac{n\pi x}{a})\Big|_{n=1}^{\infty}$ la cual se puede ortonormalizar multiplicando a cada lado de la base por $\sqrt{\frac{2}{a}}$.

Entonces, se busca una solución del problema en la forma:

$$T(x,t) = \sum_{n=1}^{\infty} C_n(t)\sqrt{\frac{2}{a}}\, sen\left(\frac{n\pi x}{a}\right) \tag{34}$$

Si la serie en 34 converge adecuadamente, la función $T(x,t)$ satisface siempre las condiciones de contorno 31.

Se expresará la función $\left(1 - \frac{x}{a}\right)$ en serie con respecto a la base ortonormal $\sqrt{\frac{2}{a}}$ sen $\left(\frac{n\pi x}{a}\right)$, con $n \geq 1$ de $L_2(0,a)$. En efecto:

$$1 - \frac{x}{a} = \sum_{n=1}^{\infty} \alpha_n \sqrt{\frac{2}{a}} \,\text{sen}\left(\frac{n\pi x}{a}\right), \tag{35}$$

donde α_n son los coeficientes de Fourier de la función $1 - \frac{x}{a}$ con respecto a dicho sistema ortonormal, es decir:

$$\alpha_n = \sqrt{\frac{2}{a}} \int_0^a \left(1 - \frac{x}{a}\right) \,\text{sen}\left(\frac{n\,\pi\,x}{a}\right)\, dx = \frac{\sqrt{2a}}{n\pi}, \tag{36}$$

Al sustituir 36 en 35, y posteriormente en 30, se llega a la ecuación [11]:

$$\frac{\partial T(x,t)}{\partial t} = \sigma\,\frac{\partial^2 T(x,t)}{\partial x^2} - \frac{2}{\pi}\,f'(t)\sum_{n=1}^{\infty}\frac{1}{n}\,\text{sen}\left(\frac{n\pi x}{a}\right), \tag{37}$$

Ahora, lo que se supone es que los coeficientes $C_n(t)$ en 34 son tales que la serie en 34 puede ser derivada término a término, una vez respecto del tiempo y dos veces respecto a x, dando como resultado, en cada caso, una función de $L_2(0,a)$ para cada valor de $t > 0$.

Así, al sustituir la expresión 34 en 37 se llega a:

$$\sqrt{\frac{2}{a}} \sum_{n=1}^{\infty} C_n'(t)\, \text{sen}\left(\frac{n\pi x}{a}\right) + \sigma \sqrt{\frac{2}{a}} \left(\frac{\pi}{a}\right)^2 \sum_{n=1}^{\infty} C_n(t)\, n^2\, \text{sen}\left(\frac{n\pi x}{a}\right) = -\frac{2}{\pi} f'(t) \sum_{n=1}^{\infty} \frac{1}{n}\, \text{sen}\left(\frac{n\pi x}{a}\right) \quad (38)$$

y agrupando de forma conveniente, se obtiene:

$$\sum_{n=1}^{\infty} \left[C_n'(t) + \sigma \left(\frac{\pi}{a}\right)^2 n^2\, C_n(t) + \frac{\sqrt{2a}}{n\,\pi} f'(t) \right] \sqrt{\frac{2}{a}}\, \text{sen}\left(\frac{n\pi x}{a}\right) = 0\,,\ \forall t > 0\,,\ \forall x \in [0,a] \quad (39)$$

Entonces, del hecho que $\sqrt{\frac{2}{a}}\, \text{sen}\left(\frac{n\pi x}{a}\right)$ una base ortonormal de $L_2(0,a)$ [13] y de 39 se deduce que:

$$C_n'(t) + \sigma(\frac{\pi}{a})^2 n^2 C_n(t) + \frac{\sqrt{2a}}{n\pi} f'(t) = 0, \quad (40)$$

Pero notar que, cada una de las ecuaciones es una ecuación diferencial ordinaria de primer orden para la función $C_n(t)$. Como además se tiene que cumplir la condición inicial 32, entonces al evaluar en $t = 0$ la expresión 34:

$$\sum_{n=1}^{\infty} C_n(0) \sqrt{\frac{n\pi\, x}{a}} = 0 \quad (41)$$

de donde, nuevamente, por ser $\sqrt{\frac{2}{a}}\, sen(\frac{n\,\pi\, x}{a})$ una base ortonormal de $L_2(0,a)$, se concluye que:

$$C_n(0) = 0 \quad \forall n \geq 1, \quad (42)$$

Si se resuelve ahora la ecuación 40 con la condición inicial 42, es fácil ver que la solución es [11]:

$$C_n(t) = -\frac{\sqrt{2a}}{n\pi} \int_0^t e^{\sigma\,(\pi/a)^2\, n^2\, (\tau - t)}\, f'(\tau)\, d\tau, \quad (43)$$

Por lo tanto, finalmente la solución $T(x,t)$ del problema (30)-(32) se expresa en la forma de la serie (34) donde los coeficientes de C_n se calculan mediante la fórmula (43).

En realidad, lo que interesa es la solución $\overline{T}(x,t)$ del problema original (14)-(18), pero esta se obtiene a partir de la solución para $T(x,t)$ utilizando la fórmula 27.

A partir de este análisis, se puede trabajar con una expresión analítica explícita para la función $f(t)$ que reproduzca condiciones de la temperatura en las habitaciones colindantes a la sala en estudio, y encontrar una expresión concreta para la solución $\overline{T}(x,t)$ del sistema, pero esto escapa del interés de este trabajo.

En esta sección se buscó describir como es el flujo de calor desde el punto de vista físico por medio de las ecuaciones de Fourier, sin embargo el trabajo matemático se encuentra disponible para su lectura en el Anexo 12.1: *Continuación del análisis matemático*.

4. Circuito de calefacción pre-existente

La **UNPSJB**[1] cuenta con un sistema de calefacción central conformado por calderas, bombas de agua, conductos de chapa galvanizada, fancoils y termostatos.

El sistema se ubica en el segundo subsuelo del edificio y esta conformado por cuatro calderas, dos de una potencia de 600 $^{kcal}/_{h}$ y dos de 700 $^{kcal}/_{h}$, que se pueden ver en la Figura 11. La función de estas calderas es calentar agua mediante un intercambio de calor con un gas combustible que circula en el interior del depósito. Comúnmente sólo se utilizan dos configuraciones, dos calderas de $^{kcal}/_{h}$ o una sola, ya sea de 600 o 700 $^{kcal}/_{h}$, según lo requieran las condiciones térmicas del día.

Figura 11: Calderas de 600 $^{kcal}/_{h}$ ubicadas en el segundo subsuelo de la universidad.

Las calderas de gas que se observan son **humotubulares** y de marca "*Niagara*", son un tipo específico de caldera industrial usada principalmente para la generación de vapor o agua caliente, y son muy eficientes para aplicaciones que requieren una alta producción de vapor, como lo es abastecer el sistema de calefacción de todo el edificio.

El funcionamiento de una caldera de este tipo consiste en la combustión de gas natural mediante un quemador ubicado en la parte inferior de cada unidad. El gas proveniente del suministro se mezcla con aire formando una mezcla inflamable que es encendida por el quemador. Esta mezcla o humo generado se hace circular por los "*tubos de humo*", observables en la Figura 12, transfiriendo el calor al agua que los rodea y circula alrededor de ellos.

[1] Universidad Nacional de la Patagonia San Juan Bosco

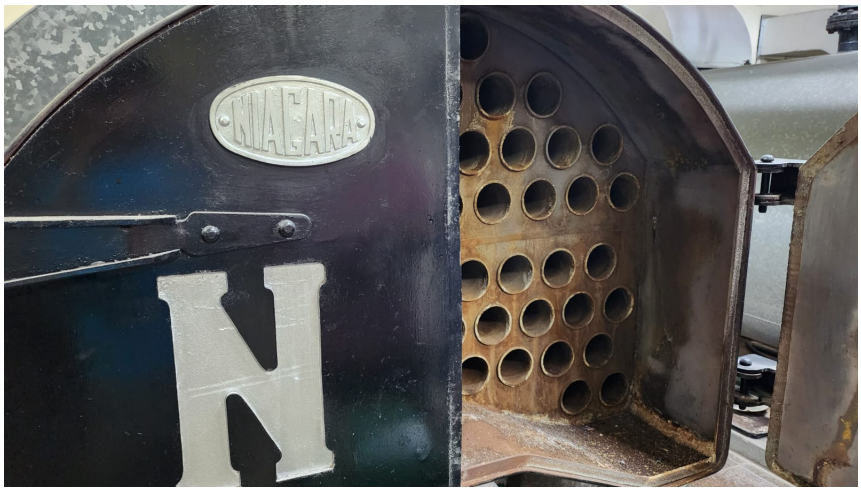


Figura 12: Interior de la caldera - Tubos de humo a la vista.

Una vez que el gas sale de la caldera se pasa a una chimenea que conecta con la azotea y allí se libera al ambiente. Por otro lado el agua, como se mencionó anteriormente, sale a mayor temperatura que a la entrada y circula hacia un sistema de bombas y tuberías, que se observan en la Figura 13.

El agua caliente se distribuye hacía los distintos circuitos de calefacción, que se dirigen a cada uno de los dos edificios de la universidad. En la Figura 13, a la derecha se observan las tuberías y bombas que suministran agua caliente al edificio principal que tiene calefacción a través de radiadores mientras que los de la izquierda son los que suministran agua caliente para la calefacción del segundo edificio, que cuenta con un sistema de calefacción basado en *fan-coils* y difusores.

En la Figura 13 se observan cuatros bombas, aunque son sólo dos las que están en funcionamiento habitualmente, mientras que las otras dos se tienen como respaldo (o "*back-up*") ante posibles fallas.

Figura 13: Sistema de bombeo

4.1. Piso del Departamento de Electrónica

La derivación que es de interés para el análisis es aquella que, a través de las cañerías, transporta agua caliente hacia el piso correspondiente al DIE[2]. El fluido caliente llega a la "Sala de Máquinas", donde se lo conecta al fan-coil central (o "ventiloconvector" central) que se observa en la Figura 14.

Este fan-coil toma aire del exterior del edificio (ver Figura 15), el cual es aspirado mediante un motor interno y calentado con un radiador interno por el cual circula el agua caliente. Una vez calentado, ese aire es el que, a través de tuberías recubiertas con material aislante térmico, se hace llegar a las distintas aulas o salas del piso.

Figura 14: Fancoil de la Sala de Máquinas

Figura 15: Entrada de aire al fancoil

Por lo tanto, el sistema de calefacción se encuentra definido de la siguiente manera:

- **Unidad central:** caldera ubicada en el segundo subsuelo, encargada de calentar el agua, tal como se describió previamente.
- **Distribución:** el agua caliente se distribuye gracias al sistema de bombeo del subsuelo y las tuberías que recorren el edificio.
- **Fan-coil:** ubicado en la sala de máquinas, es el encargado de transferir calor al del agua caliente al aire proveniente desde el exterior.
- **Distribución del aire:** El aire caliente se distribuye a través de conductos por el piso del departamento.

[2]Departamento de Ingeniería Electrónica

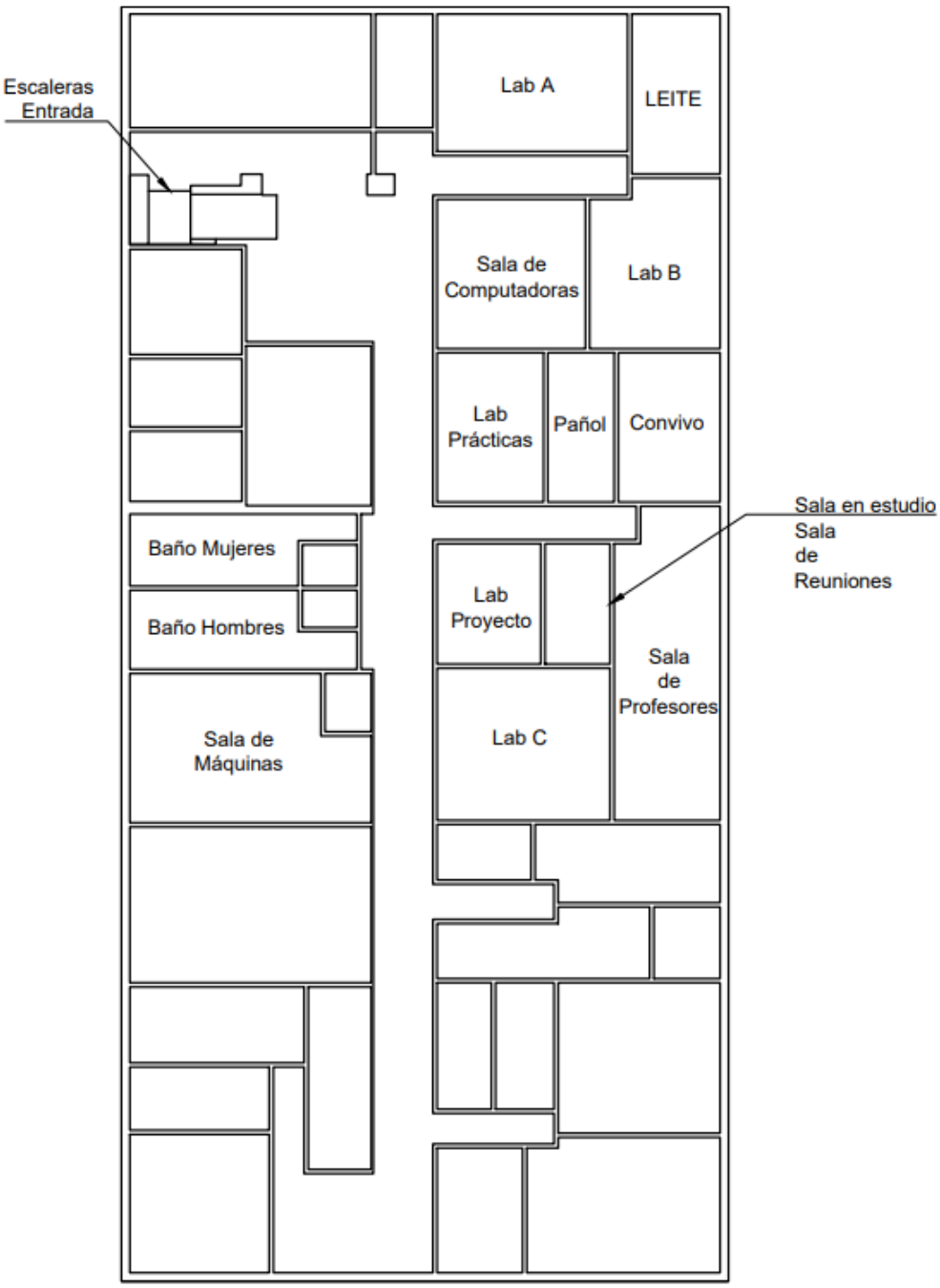


Figura 16: Plano del Departamento de Electrónica

En la Figura 16 se puede observar el tercer piso del segundo edificio de la universidad, donde se encuentra el DIE.

En la Figura 17 se observa el conducto principal, proveniente del fan-coil, desde el interior de la sala de máquinas, como sale de la misma casi a la altura de la puerta de la sala. Siguiendo el conducto principal, este se divide en dos ramas principales que pueden observarse en la Figura 18. La rama de interés es la de la derecha, que es la que se corresponde con la zona donde se encuentran las aulas y laboratorios del DIE,

la otra va hacia la izquierda, donde se encuentran aulas y laboratorios de la Facultad de Ciencias Naturales.

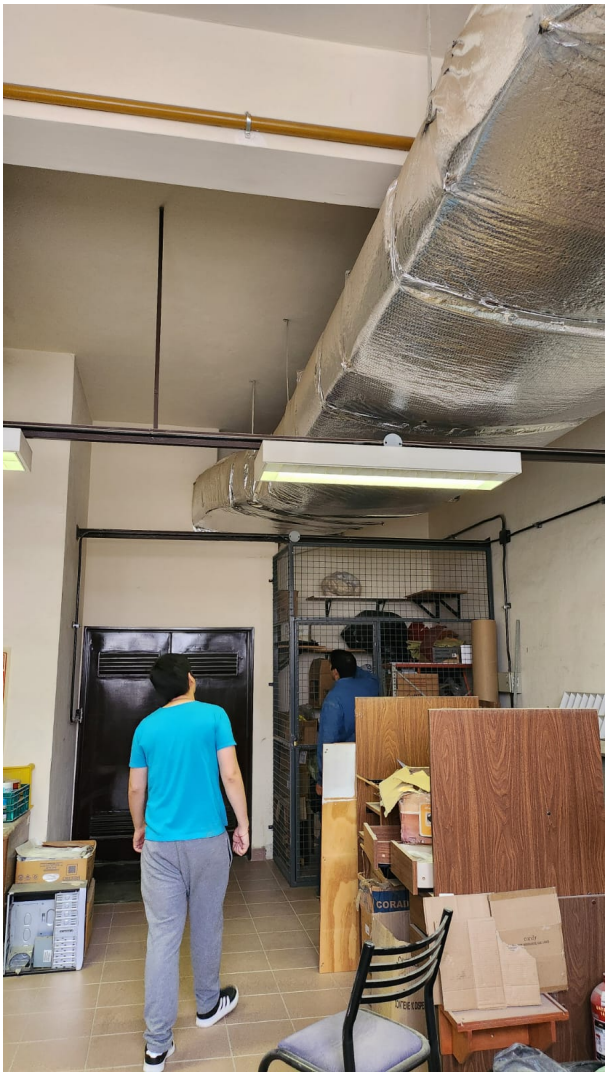


Figura 17: Conducto principal visto desde el interior de la sala de máquinas.

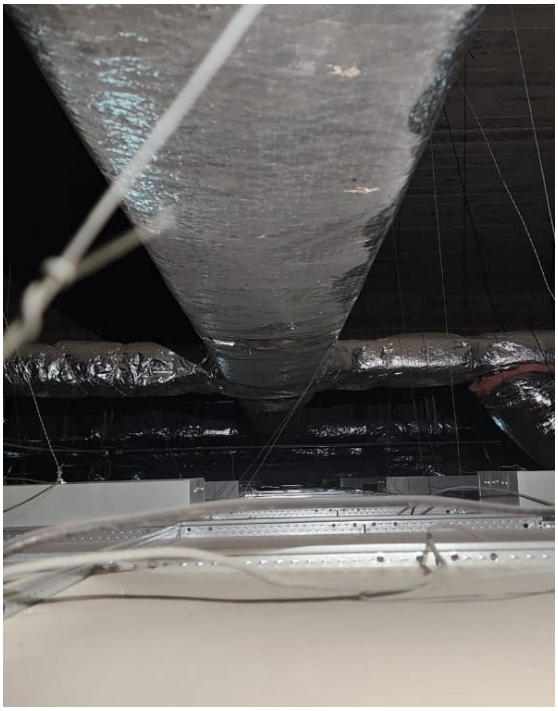
A partir de este punto, se procede a analizar el recorrido del circuito de calefacción correspondiente al DIE, haciendo especial énfasis en cómo llega el flujo de aire hasta la sala de reuniones. Esto se realiza con el objetivo de estimar el caudal de aire que ingresa a la sala de estudio, y poder contemplarlo adecuadamente en el modelo matemático. Resulta imprescindible para que el modelo represente lo más fielmente posible la acción de control que se tendrá, siendo la acción de control el flujo de aire que ingresa en la habitación.

Para analizar el recorrido, se realiza un relevamiento visual de los ductos de ventilación, observando su distribución por encima del cielorraso a lo largo del piso. En la Figura 19a se puede observar un ducto secundario que va a varias aulas del DIE y como se separa en otros dos que culminan en dos difusores. Dependiendo el tamaño del aula suele tener dos o cuatro difusores.

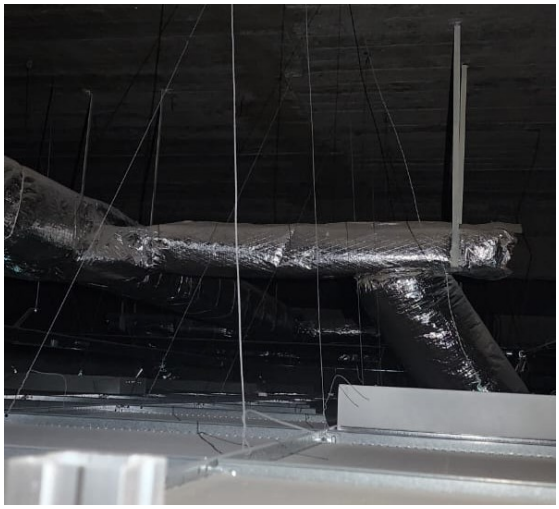
Figura 18: Bifurcación principal, a la derecha hacia el DIE.

Siguiendo todo el circuito, se puede identificar el final del recorrido, que desemboca en la sala de reuniones, se puede observar en la Figura 19b.

La sala en estudio, como ya se ha mencionado anteriormente, es la *Sala de Reuniones*, marcada en el plano mostrado en la Figura 16. A partir de esta imagen, se marca el recorrido de los conductos desde la sala de máquinas hasta la de reuniones en la Figura 20. El tamaño de las líneas respeta como van disminuyendo el tamaño de los conductos con cada ramificación.



(a) Conductos de aire sobre las aulas



(b) Final del recorrido a la Sala de Reuniones

Figura 19: Conductos de aire sobre las aulas.

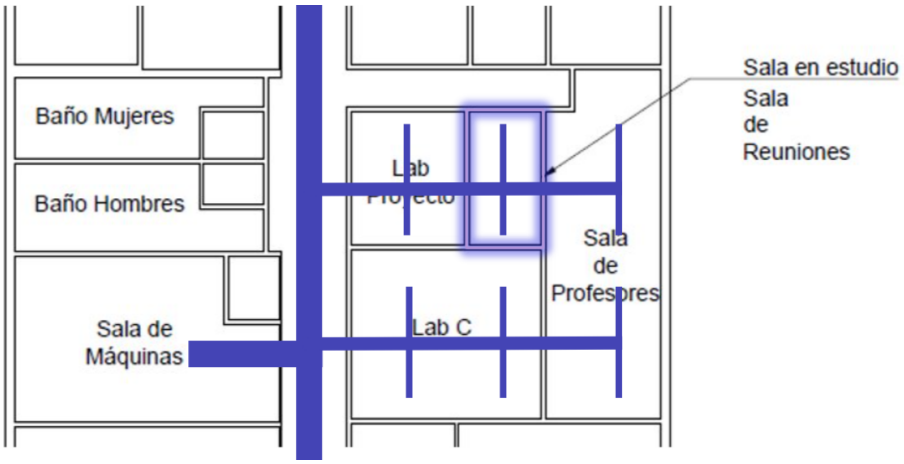


Figura 20: Conductos hasta la zona de interés

4.2. Difusores actuales

El componente final del recorrido de la instalación preexistente son los difusores de aire. Los hay de dos tipos, circulares y cuadrados, como se puede ver en las Figuras 21a y 21b. Estos permiten o restringen el paso del aire hacia las habitaciones y su grado de apertura determina el caudal de aire que ingresa a la habitación. El difusor constituye entonces el actuador del sistema planteado en la Figura 2.

El sistema de control regulará el flujo de aire que permiten los difusores al modificar su grado de apertura, en función de la temperatura que el usuario desea y la temperatura a la que se encuentra la planta.

(a) Difusores circulares

(b) Difusores cuadrados

Figura 21: Tipos de difusores instalados actualmente en el edificio.

Los difusores instalados en la sala de reuniones son de tipo circular y pueden observarse más detalladamente en la Figura 22. El principal defecto de estos artefactos

radica en la dificultad que presentan para ajustar su apertura y cierre, lo cual reduce su operatividad práctica.

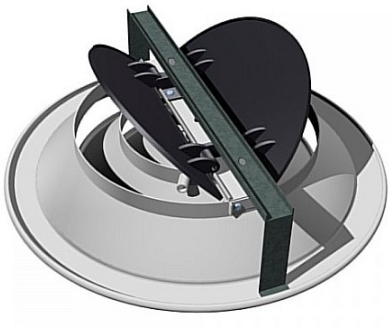


Figura 22: Modelo de difusor circular actual con su mecanismo de apertura

Debido a lo incómodo que resulta el ajuste, en la práctica, los usuarios se ven obligados a adaptarse a las condiciones térmicas del ambiente, teniendo así un uso ineficiente del sistema y desperdiciando energía térmica.

4.3. Analogía eléctrica

Realizando una analogía eléctrica con los sistemas térmicos, se puede representar cada habitación como un conjunto R-C, ya que almacena calor (C), y presenta una oposición inherente al flujo de calor (R). El caudal de aire, la salida del fancoil, se relaciona con la corriente eléctrica y las temperaturas con las diferencias de potencial.

Se construye entonces un circuito como el de la Figura 23 para representar esta analogía, con la suposición de que las resistencias R son todas iguales. Se reparte entonces de forma equitativa en cada nodo, lo que facilita la estimación del caudal que ingresa a la habitación, siendo este de $\frac{V_{fancoil}}{8}$.

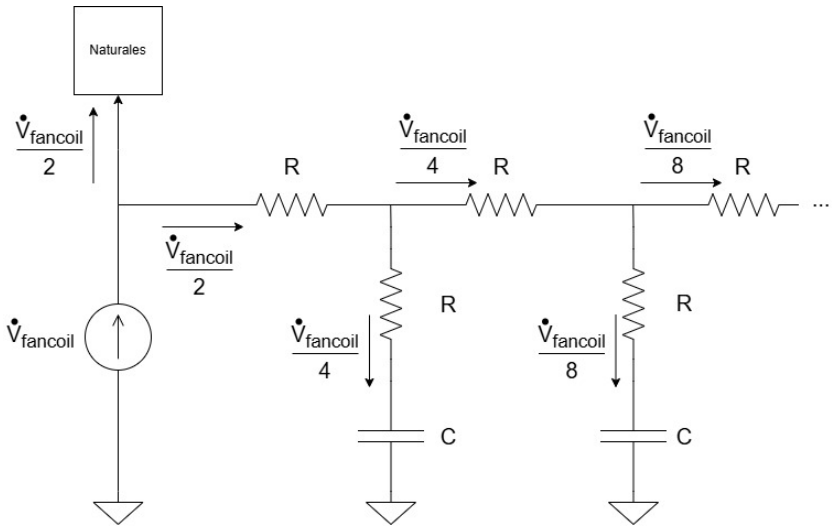


Figura 23: Circuito análogo al sistema térmico bajo estudio

La hoja de datos del fancoil se encuentra en el Anexo 14.1: *Fancoil del piso de electrónica - Marca Carrier*, de allí se obtiene el dato del caudal de salida del mismo, siendo este de:

$$\dot{Q}_{fancoil} = 10{,}200\, m^3/h$$

Por la estimación del circuito de la Figura 23, el caudal que iría a la sala en estudio sería una octava parte de este valor

$$\dot{Q}_{fancoil} = 1700\, m^3/h$$

cuando los difusores se encuentran abiertos al máximo.

5. Mediciones, ensayos y modelado

El modelado de un sistema implica obtener una representación matemática del mismo, a partir de la información que se pueda recopilar. Generalmente, existen dos formas de modelado, una se basa en la interacción con el sistema y la otra en el análisis físico del mismo [6].

El primero de los dos casos es el que se desarrolla en este trabajo, considerando el sistema como una caja negra, de la cual se tiene conocimiento de las entradas y salidas pero no se tiene un **conocimiento total** de lo que sucede dentro. Para señales de entrada conocidas, un escalón en este caso, se miden las salidas y se obtiene un modelo del sistema. Una vez se realizan los ensayos que determinan la respuesta de la planta ante un escalón de entrada, se utilizan métodos de identificación de sistemas para hallar un modelo que se adecúe a dicho comportamiento.

Este modelo, que se obtiene analíticamente, se utiliza tanto para interpretar el comportamiento de la planta bajo estudio como para diseñar el controlador que comande el **funcionamiento deseado**.

5.1. Ensayos

Para caracterizar la planta, se realizan una serie mediciones de temperatura de la habitación a lo largo del día. Este proceso se realiza durante varios días siguiendo los siguientes pasos:

1. El día previo a realizar el ensayo, se tapan las salidas de aire con un cartón y un material con buena absorción térmica (ver Figura 24).

Figura 24: Salidas de aire selladas.

2. En el día del ensayo, se aplica el escalón de caudal de aire correspondiente encendiendo el fanciol. Esto se realiza mediante la configuración del termostato que se puede observar en la Figura 25.

Figura 25: Termostato que controla el fancoil de la sala de máquinas.

3. Se espera a que el flujo de aire se establezca por el circuito de calefacción. Con la ayuda de un anemómetro de la marca UNI-T, como el de la Figura 26, se verifica ésto.

Figura 26: Anemómetro utilizado para medir el caudal de aire que circula por los ductos.

4. Una vez establecido el flujo de aire en los conductos, se destapan las salidas de aire, provocando que el aire caliente la sala de Profesores.

5. Utilizando un sensor de temperatura **DS18B20**, y una Rasberry Pi Pico, se mide la temperatura en la sala de profesores (Figuras 27 y 28).

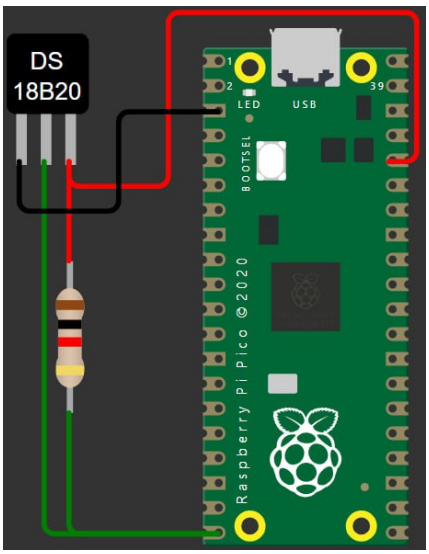


Figura 27: Esquema de conexión del sensor conectado a la Raspberry Pi Pico W

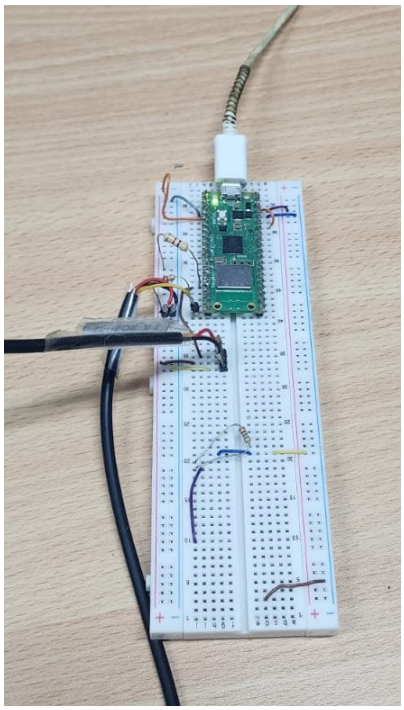


Figura 28: Implementación en la sala de profesores

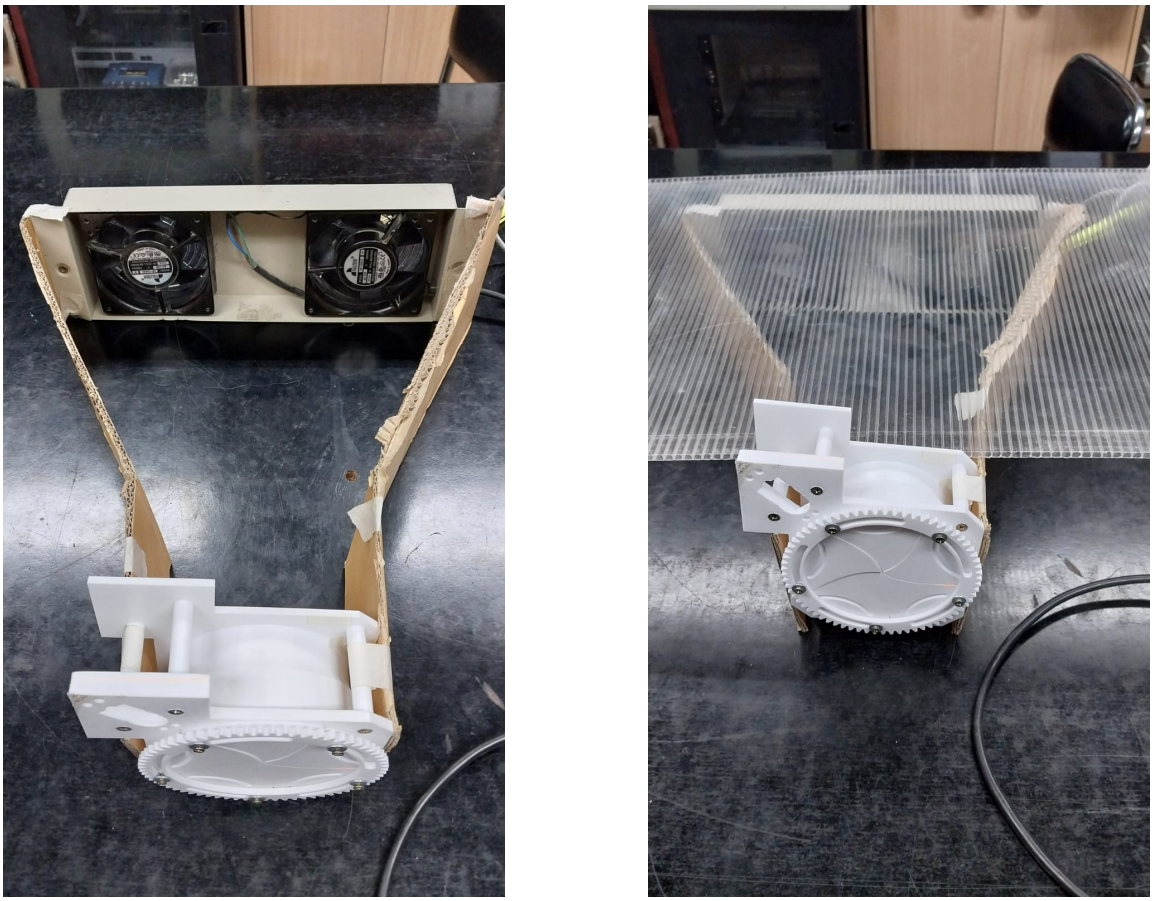
6. Se recopilan los datos en *ThingSpeak*[3] para posterior análisis y observar como se comporta la planta frente al escalón de entrada aplicado.

5.1.1. Sensor de temperatura

El sensor que se utiliza es el que se observa en la Figura 29, el **DS18B20**. Este utiliza el protocolo *1-Wire para comunicarse*, el cual sólo necesita de un pin de datos para comunicarse y permite conectar más de un sensor en el mismo bus de datos.

Figura 29: Sensor digital DS18B20

[3]Plataforma de servicios de IoT de MathWorks que permite agregar, visualizar y analizar flujos de datos en vivo en la nube. Disponible en: https://thingspeak.com/



(a) Vista superior de la maqueta (b) Maqueta cerrada

Figura 30: Maqueta preparada para los ensayos.

5.2. Curva de funcionamiento del difusor

Se evalúa el comportamiento del difusor diseñado ante variaciones en su grado de apertura, se busca poder observar la relación entre el caudal y el grado de apertura para poder observar cómo es el comportamiento del actuador.

Con ese objeto, se prepara la maqueta que se muestra en las Figuras 30a y 30b. En ella se utilizan coolers como fuentes de aire, que trasladan aire a través de un espacio cerrado al difusor, con el fin de simular condiciones óptimas para observar el comportamiento del caudal a la salida con el grado de apertura. Más detalles sobre el difusor que aparece en las Figuras 30a y 30b se encuentran próximamente en el Capítulo 7.7: *Segunda etapa: Rediseño*.

Se vuelve a utilizar el anemómetro UNI-T presentado anteriormente, de la forma que se ve en la Figura 31 y se registran los valores con con la aplicación de UNI-T, obteniendo la curva de la Figura 32.

La apertura del difusor se fue variando gradualmente, de esta forma se obtiene un valor de velocidad de caudal para distintos tiempos, correspondiéndose cada tiempo a un grado de apertura. La curva obtenida permite ver una región aproximadamente

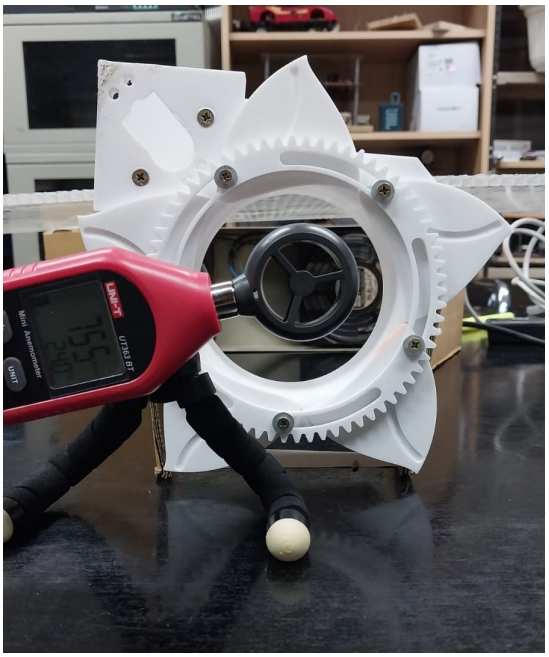


Figura 31: Anemómetro para medición de velocidad a la salida

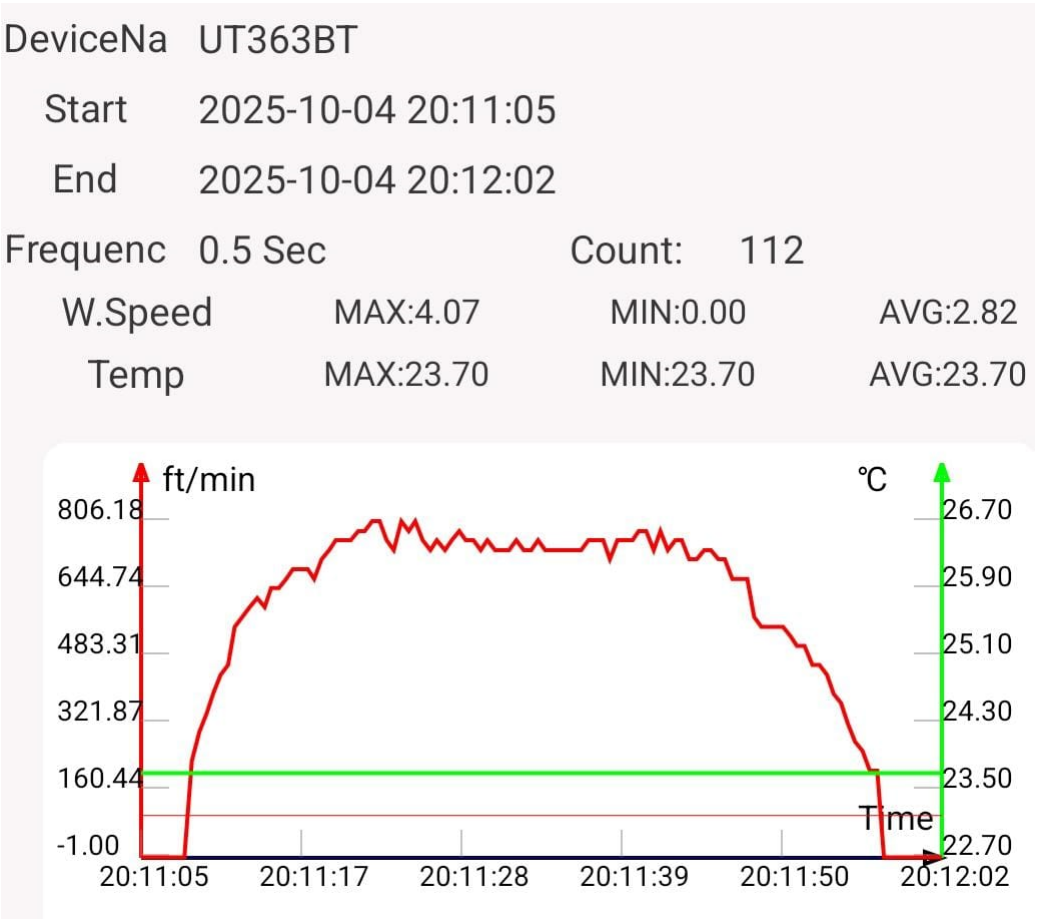


Figura 32: Curva obtenida mediante el ensayo

lineal, en la cual la velocidad del aire se comporta de manera proporcional al grado de apertura, esto sucede para los grados de apertura entre el 20 % y 70 %.

5.3. Obtención de modelo

La mayoría de los sistemas térmicos observados en la naturaleza, obedecen una dinámica de primer orden [4]. En estos casos, las funciones de transferencia se cumplen con la forma:

$$\frac{\theta(s)}{H_i(s)} = \frac{1}{RCs + 1} \tag{44}$$

donde $\theta(s)$ es la transformada de Laplace de la temperatura conforme varía el tiempo, y $H_i(s)$ representa la entrada de flujo de calor. A su vez, R y C están asociadas a la dinámica del sistema.

En el caso de este sistema, la entrada al mismo es el caudal de aire caliente que circula por los conductos, el cual se puede estimar a partir de los datos del fancoil y de las dimensiones de los conductos. El caudal estimado se contrasta con una medición realizada con un anemómetro.

Una vez se obtienen los datos de los ensayos, se procede a realizar una identificación de sistemas, a través del método gráfico introducido en la sección del marco teórico. Las curvas obtenidas mediante los ensayos realizados, tienen justamente la apariencia de representar un comportamiento de primer orden (una respuesta sigmoidea), de modo que se presenta el modelo FOPDT:

$$G(s) = \frac{K_P\, e^{-Ls}}{1 + T\, s} \tag{45}$$

Se registra el horario en el cual se le aplicó el escalón al sistema, el cual representa el t_0, para poder obtener el retardo que presenta el sistema, es decir el tiempo que tarda la salida en responder al escalón aplicado.

En el primer ensayo realizado se aplica un escalón del 0 % al 100 %, es decir que en un instante inicial el difusor está completamente cerrado y una vez la temperatura se establezca en un valor, se lo abre por completo y se observa la dinámica de la salida. El resultado obtenido se puede observar en la Figura 33.

Aplicando el método, teniendo en cuenta que el escalón aplicado es el caudal máximo que puede circular por la salida de aire ($\dot{Q}_{in} = 1700\, m^3/h$), e identificando los distintos parámetros que permiten obtener el modelo de la planta:

Lo cual permite calcular:

$$K_p = 0,00294$$
$$T = 900\ seg$$
$$L = 360 seg$$

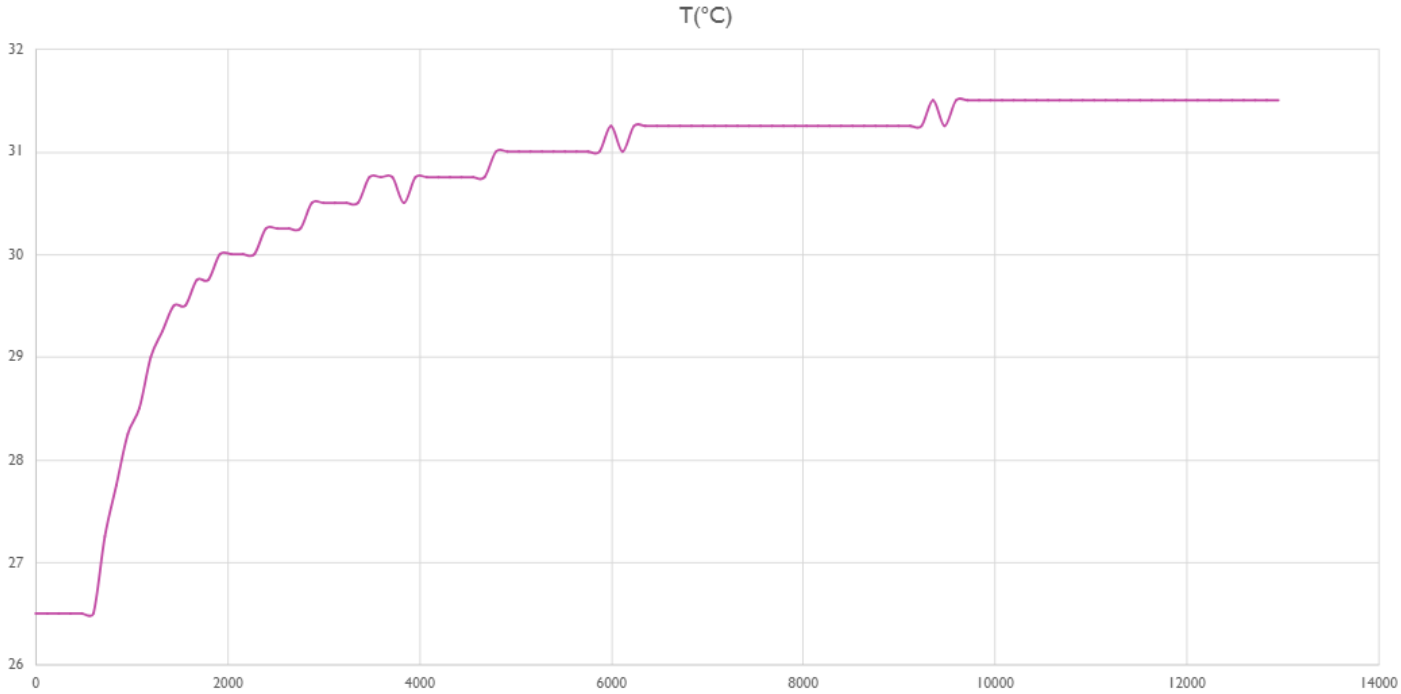


Figura 33: Respuesta de la planta antes un escalón de 0 a 100

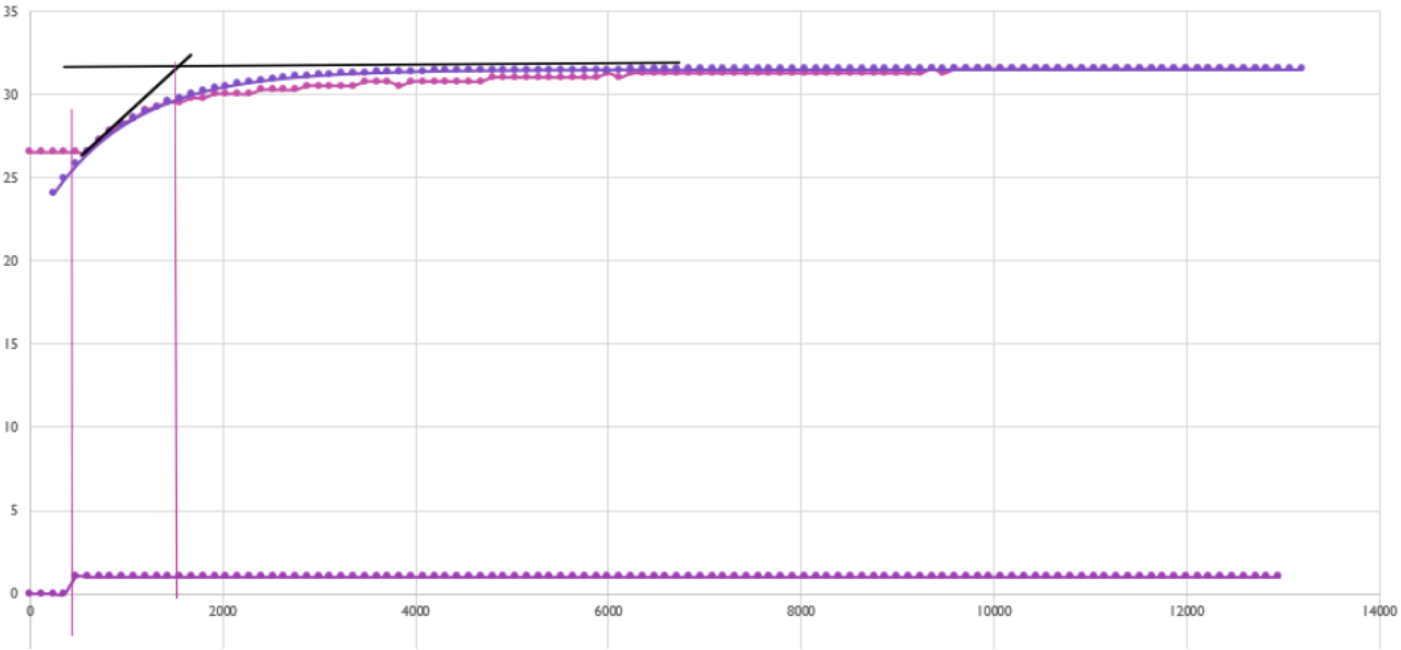


Figura 34: Identificación de parámetros

Por lo tanto, la planta obtenida es:

$$G(s) = \frac{0,00294}{1 + 900\,s} e^{-360s}$$

En la Figura 34 se observa este modelo con la gráfica azul, mientras que en rosa se encuentran los datos del ensayo. Se verifica que el modelo calculado se ajusta muy bien a los datos, por lo que es un modelo que se podría utilizar.

Ahora bien, hay un inconveniente en todas estas mediciones realizadas, que tiene que ver con la naturaleza no lineal del actuador, y es que esta se present en forma de una "Saturación", es decir que el flujo de caudal que circula por él es proporcional al grado de apertura solamente en un intervalo determinado de apertura del difusor.

Cuando el difusor se encuentra parcialmente cerrado (grados de apertura bajos) el caudal es muy pequeño, y al abrirlo a aproximadamente un 75 % se observa que llega a un límite en el caudal que por el circula, es decir que si se abre más, no se observan grandes diferencias en el caudal. Este fenómeno se puede apreciar mejor en la Figura 35, donde se aplica un escalón de 0 % a 100 % en distintos puntos de operación, y se observa que el sistema responde con distinta constante de tiempo, lo cual es evidencia de un sistema no lineal, por lo que no se puede tratar el problema con las técnicas vistas en las materias de Control.

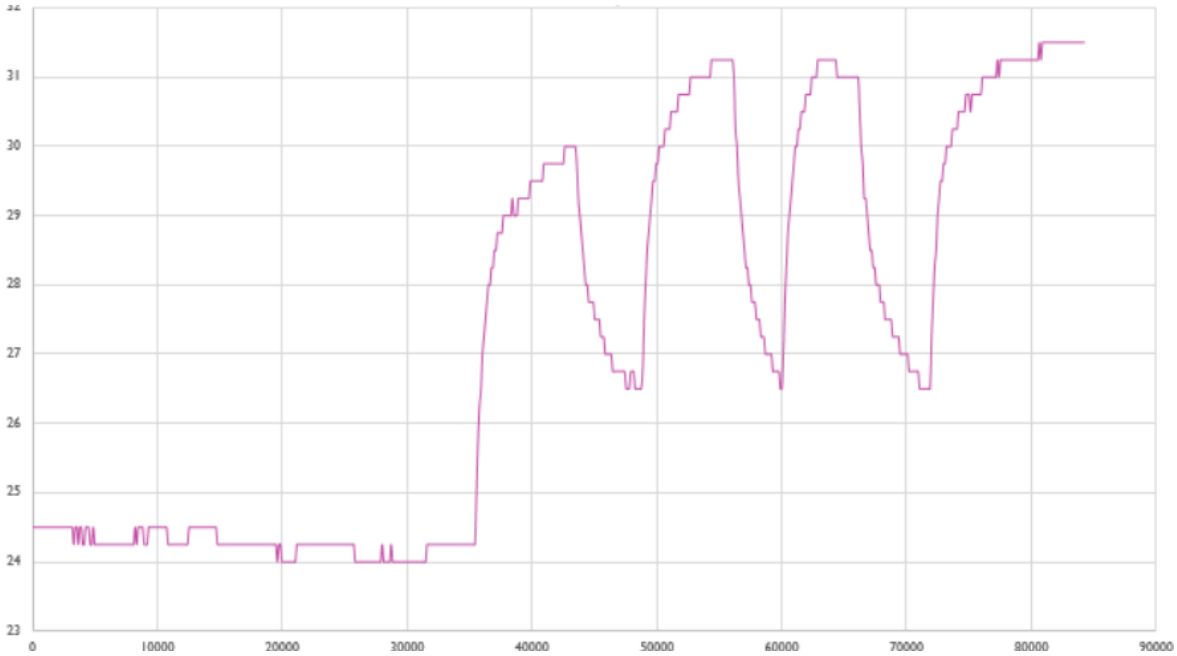


Figura 35: Respuesta ante la aplicación del escalón en distintos puntos de operación

Al ser un sistema no lineal, el modelo que se obtiene sólo es válido en un entorno de trabajo, alrededor de la apertura completa, pero no es válido para el resto, por lo que con esta técnica de abrir desde el 0 % en adelante se obtienen funciones de transferencia que son válidas en entornos pequeños, por lo que a cada planta calculada le correspondería un controlador específico.

Ahora bien, existe un intervalo en que el actuador se comporta de forma lineal, por lo que si se realizan ensayos en esta zona, la función de transferencia que se obtiene es válida en ella. Se realizan entonces, los ensayos entre el 20 % y el 70 %, y la función de transferencia que se obtiene describe bien el comportamiento en esa zona y el controlador calculado también será apropiado para este rango.

Por lo tanto, para obtener el modelo que se utilizará en el trabajo, se realizan ensayos para grados de apertura que aseguren que el actuador se encuentra en su zona lineal de funcionamiento. De esta forma, el modelo obtenido podrá ser utilizado con las técnicas de control vistas (control clásico aplicado a sistemas LIV).

Para ello, se abre el difusor aproximadamente al 20 % y se deja evolucionar hasta que la temperatura se establezca en cierto valor. Una vez establecido, se aplica un escalón del 20 % al 70 % de apertura, y se observa cómo es la dinámica de la planta. Con la dinámica observada, se procede a calcular la planta de forma idéntica a cómo se realizó anteriormente.

En esta medición, se discriminan:

$$K_p = 0,00235$$
$$T = 500\ seg$$
$$L = 240seg$$

De modo que la planta obtenida finalmente es:

$$G(s) = \frac{0,00235}{1 + 500\,s} e^{-240s}$$

la cual no dista mucho de la calculada anteriormente, pero que esta vez no se contempla la no-linealidad que introduce el actuador.

6. Control

La variable controlada es la temperatura de la habitación, mientras que la acción de control está dada por el volumen de aire por unidad de tiempo que ingresa a la habitación. Por lo general, los sistemás térmicos son de dinámica lenta, y su control no es tan demandante en términos de velocidad (cómo podría ser controlar algún sistema de origen electrónico). El objetivo es realizar un control con las técnicas de control clásico, desarrolladas en Control I.

Una vez ya modelado el sistema (expresión que se encuentra en la sección de modelado), se procede a calcular un controlador Pi (con las reglas de Ziegler-Nichols y las de Cohen-Coon), y analizar cómo es la estabilidad del sistema completo, así com también se discretiza el controlador calculado para finalmente implementar la ecuación de diferencias en el microcontrolador. Al tratarse de un sistema SISO, se trabaja exclusivamente con la función de transferencia que relaciona la temperatura de salida con el caudal de aire ingresante.

6.1. Control continuo

En este proyecto se aplica un control PI, que asegura una implementación simple y eficiente (si se sintonizan de forma correcta los parámetros del controlador). El diseño del mismo se basa en el modelo de primer orden, que se obtuvo mediante los métodos gráficos.

Se analiza el sistema mediante el estudio de las respuestas al escalón y los diagramas de polos-ceros para poder verificar cuestiones de estabilidad, de error en estado estacionario,etc.

6.1.1. Diagrama de polos y ceros

Es posible visualizar el efecto de la variación de la planta al analizar en primera instancia el diagrama de polos y ceros del sistema. Conocer el diagrama de polos y ceros de la planta permite interpretar la respuesta temporal, para dar lugar al controlador que se debe realizar. En la Figura 36 se representa el diagrama de polos y ceros de la planta $G(s)$.

Se puede apreciar que la planta posee un solo polo real en el semiplano izquierdo, verificándose que no existan polos en el semiplano derecho, lo que representaría un un comportamiento inestable, se dice entonces que la planta es inherentemente estable.

Si bien la no linealidad no es modelada, esta misma (al tratarse de una saturación) introduce en el lazo un polo de frecuencia muy elevada, pero el polo de la planta es de mucha menor frecuencia, por lo tanto este último será quién domine la respuesta, y el del actuador podría despreciarse en el análisis.

En función de lo analizado, al tener un único polo real, la respuesta al escalón ascendería según la constante de tiempo y se establecería en un valor de temperatura, siguiendo una forma sigmoidea propia de los sistemas de primer orden.

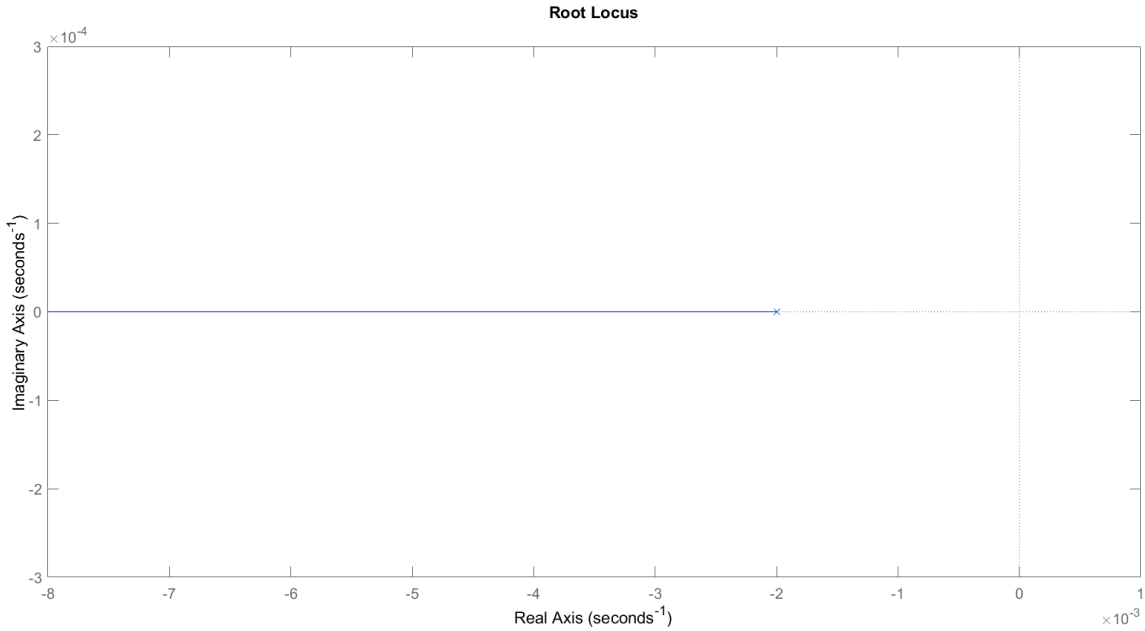


Figura 36: Lugar de raíces $G(s)$

6.1.2. Respuesta al escalón e influencia del retardo

Si se observa la respuesta al escalón de la planta a lazo abierto, se puede apreciar cómo en 500 s llega al 63 % del valor final, lo cual se observa en la Figura 37.

Ahora bien, el sistma en estudio presenta un retardo de 240 s, por lo que se debe de evaluar el impacto del mismo, y en la Fig.37 no se lo configuró. Entonces, si se considera que un escalón de la acción de control es aplicado en $t = 0$, la presencia del retardo (Fig.38):

En un diagrama de lugar de raíces, no se observa ningún efecto del retardo (sólo se podría dibujar el lugar de raíces contemplando el retardo, mediante la aproximación de Padé, que introduciría polos y ceros según el orden de la misma). Donde si es posible ver su influencia es en la respuesta en frecuencia (Fig.39), para lo cual si se observan los diagramas de Bode:

Se puede ver la influencia del retardo en el diagrama de fase, si se aprecia el rango de frecuencias observado, se ve que la ganancia es siempre menor a 0 dB, y que el desplazamiento de fase que introduce el retardo es considerablemente grande. Se sabe que la inestabilidad ocurre en la frecuencia donde el diagrama de amplitud cruza los 0 dB, y como siempre está por debajo, se dice que el sistema es intrínsecamente estable, independientemente del comportamiento de la fase.

6.1.3. Controlador continuo

Para el diseño del controlador, se realiza un controlador PI, con la idea de luego discretizarlo. El diagrama de bloques de la planta $G(s)$ y el controlador se presenta en la Fig.40

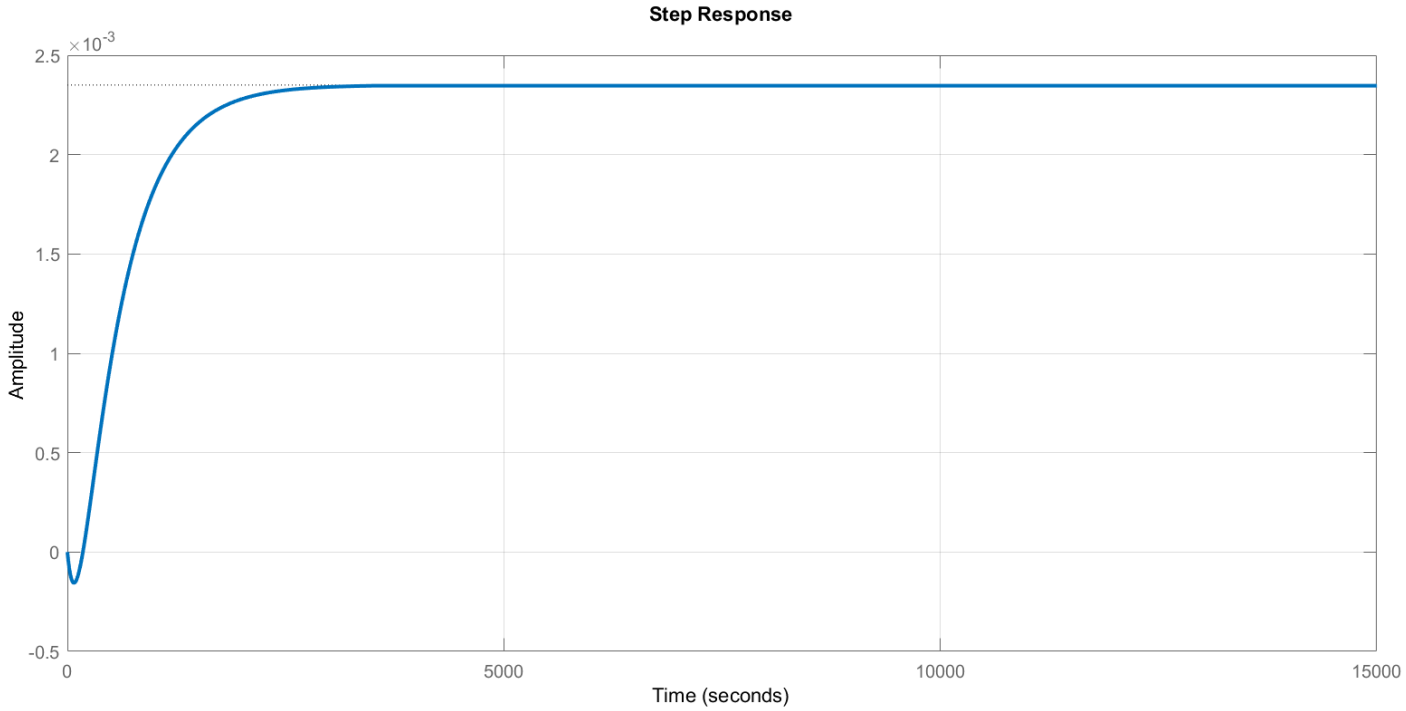


Figura 37: Respuesta al escalón $G(s)$

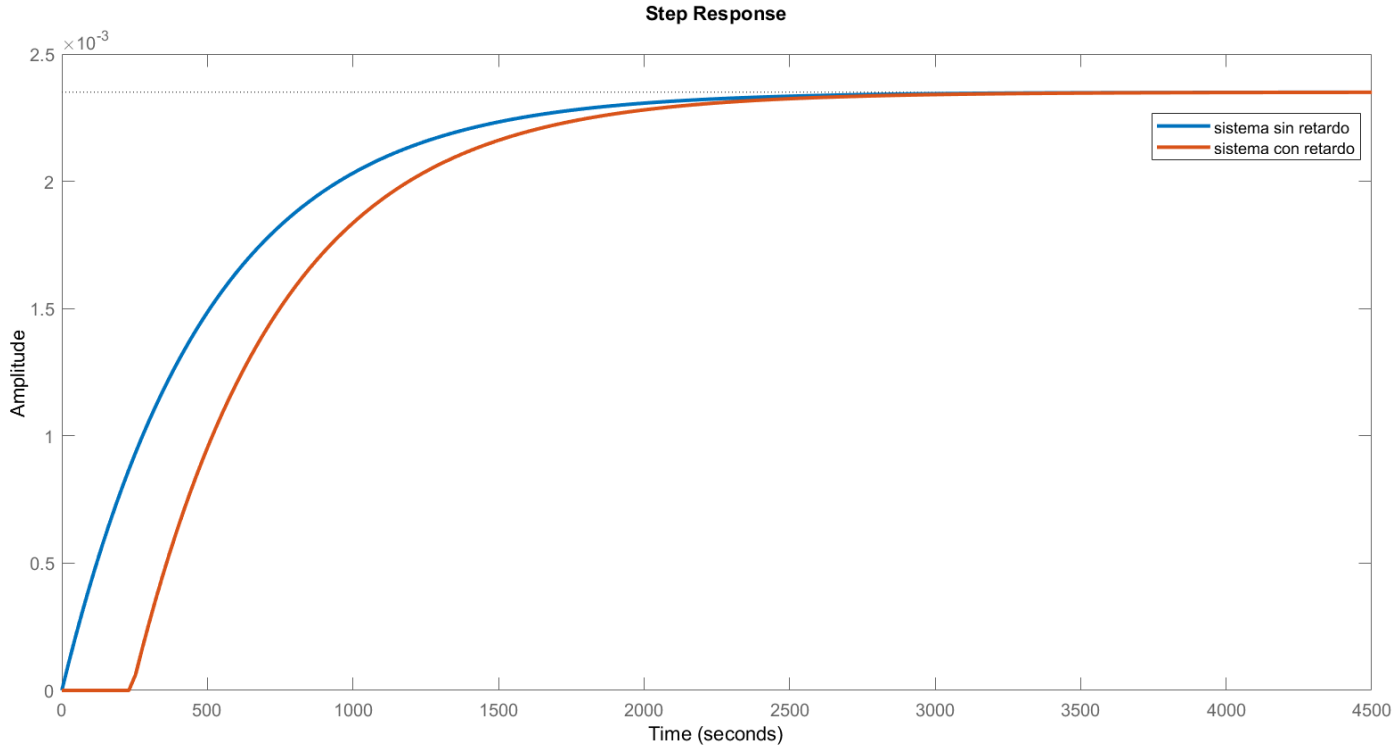


Figura 38: Respuesta al escalón $G(s)$ - Con retardo (naranja) y sin retardo (azul)

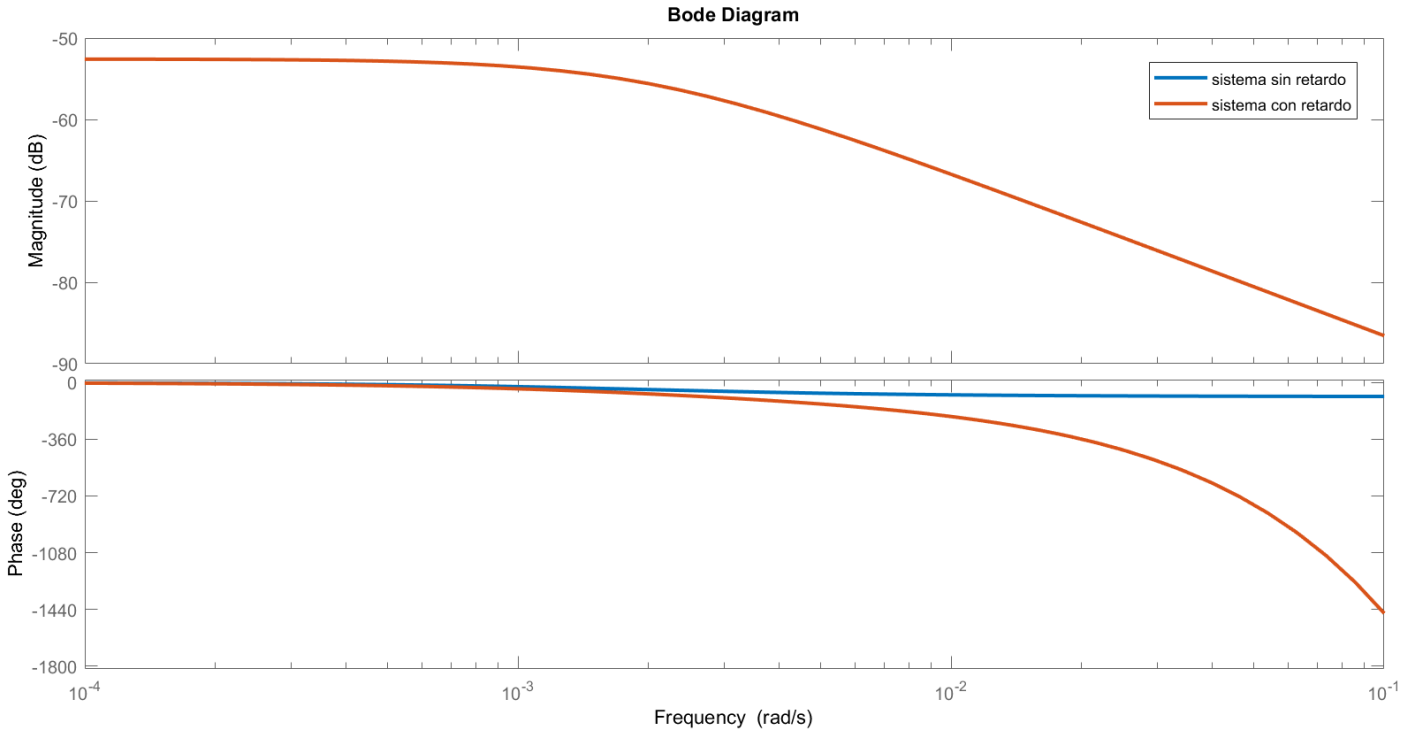


Figura 39: Diagrama de Bode - Sistema con y sin retardo

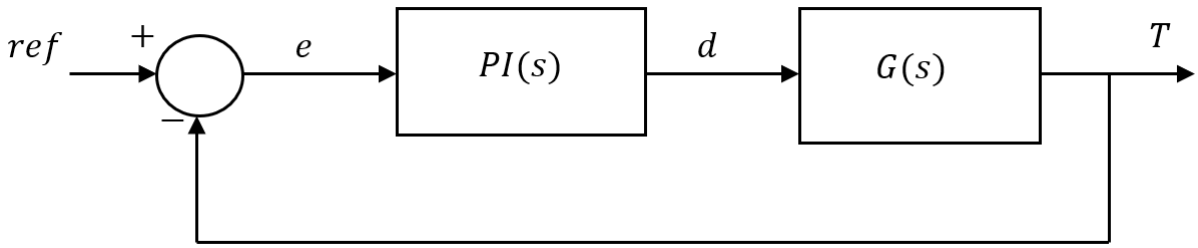


Figura 40: Lazo cerrado $G(s)$ con controlador $PI(s)$

Se diseña un controlador PI con el fin de poder aprovechar la acción integral (el polo en el origen) para poder reducir al máximo el error de estado estacionario, y cualquier variación/perturbacón aque se pueda introducir en el lazo y depende de factores externos a lo que abarca el control propuesto (como pueden ser ventanas abiertas, flujo de gente en la habitación y cualquier factor no deseado que involucre una variación en la temperatura medida). Aparte, la acción de control proporcional permite ajustar el margen de ganancia y de fase para poder asegurar la estabilidad de todo el sistema. La función de transferencia del controlador PI, al cual se le desean ajustar las constantes K_P y T_i tiene la forma:

$$PI(s) = K_P \left(1 + \frac{1}{T_i\, s}\right) = \frac{K_P\, T_i\, s + K_P}{T_i\, s} = \frac{K_P}{T_i} \frac{\left(\frac{s}{1/T_i} + 1\right)}{s} \tag{46}$$

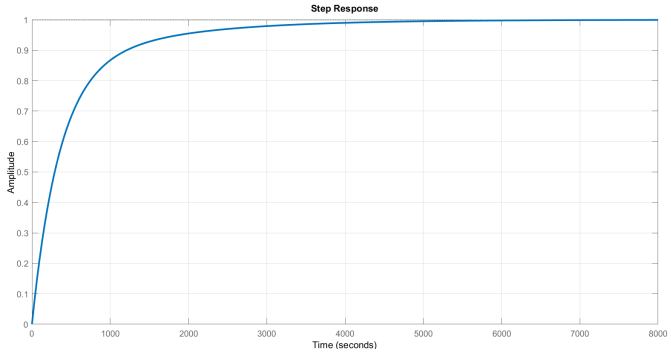


Figura 41: Lazo cerrado con controlador sintonizado con ZN sin retardo

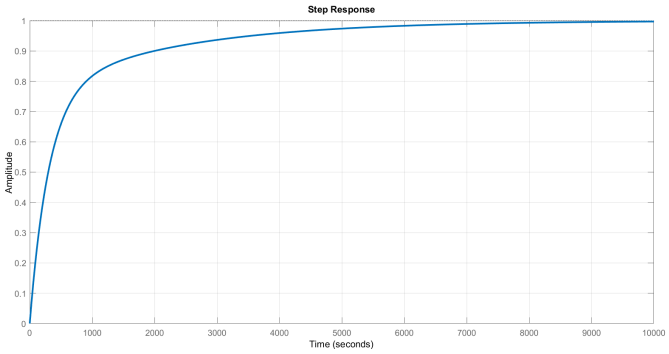


Figura 42: Lazo cerrado con controlador sintonizado con CC sin retardo

Si se realiza una sintonización inicial, mediante el método emírico de Ziegler Nichols, se obtiene el controlador:

$$PI(s) = 1{,}87\left(1 + \frac{1}{720\,s}\right) \tag{47}$$

Mientras que con la sintonización de Cohen-Coon, se obtiene el controlador:

$$PI(s) = 1{,}95\left(1 + \frac{1}{405{,}67\,s}\right) \tag{48}$$

Al tener un retardo relativamente grande (respecto a la constante de tiempo de la dinámica de la planta), se opta por usar la sintonización Cohen Coon.

6.1.4. Análisis completo

Ya con el modelo matemático de la planta y el controlador calculado, se procede a realizar un análisis de lazo cerrado, insertando el controlador (Fig.**??**), realizando un análisis tanto con retardo cómo sin retardo, para analizar como se comporta a lazo cerrado.

En primer instancia, se aplica un escalón a la entrada del lazo, y se evalúa la respuesta del sistema sin tener en cuenta el retardo de la planta:

Se puede observar en las Figuras 41 y 42 que en el caso de la sintonización con las reglas de Ziegler Nichols tarda un tanto más en establecerse en el valor final respecto a la sintonización con las reglas de Cohen-Coon. Esto se compensa en que, con la presencia de un retardo, existe un menor sobreimpulso en la respuesta del lazo que contiene el controlador calculado con la sintonización de Cohen Coon, tal como se puede observar:

Ahora bien, en el sistema en estudio, el retardo puede variar. El retardo que se tiene en cuenta, es cuando el sistema de calefacción de la Universidad opera de manera normal, pero este retardo puede ser mayor debido a que hay veces en que el fancoil está sometido a rutinas de parada, en dónde al encender el termostato, tarda un poco más

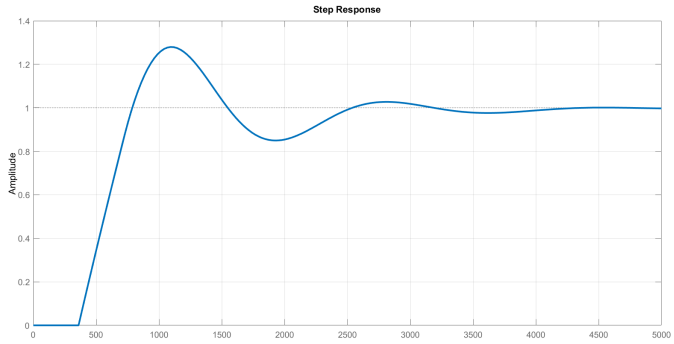


Figura 43: Lazo cerrado con controlador sintonizado con ZN con retardo

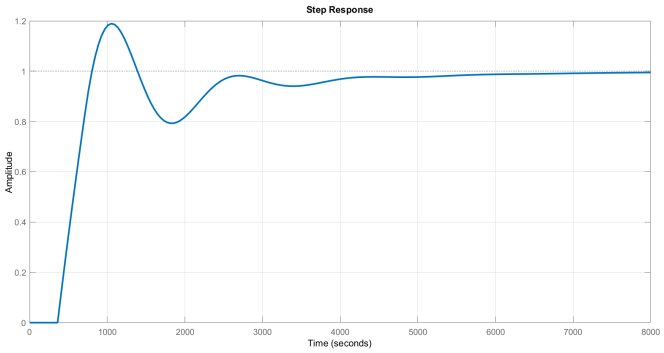


Figura 44: Lazo cerrado con controlador sintonizado con CC con retardo

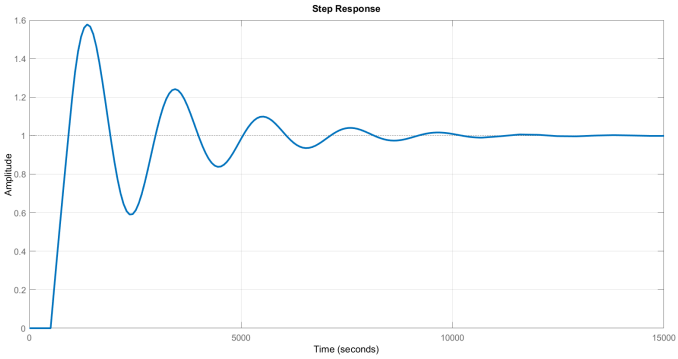


Figura 45: Lazo cerrado con controlador sintonizado con ZN con retardo $L = 500\,seg$

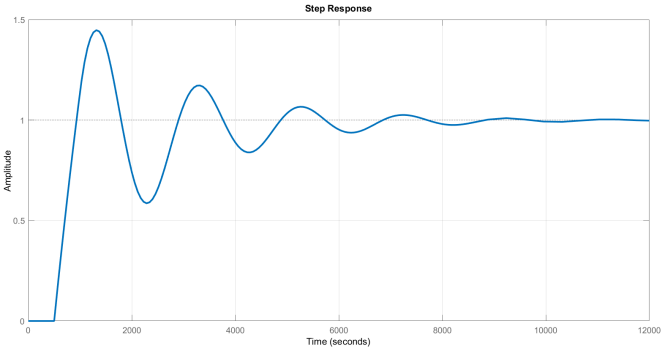


Figura 46: Lazo cerrado con controlador sintonizado con CC con retardo $L = 550\,seg$

en distribuir el aire por los ductos, y no hay posibilidades de que se pueda controlar esto (es externo al sistema).

La presencia de un retardo en un lazo de control es un factor desestabilizador y limitante para el diseño de un sistema. Cuando se cierra el lazo incluyendo la planta y el controlador, cualquier retardo en la medición o en la actuación del sistema tiene ciertas influencias principales:

- Influencia crítica en la estabilidad: introduce un retraso de fase adicional que aumenta linealmente con la frecuencia.
- Aumento del sobreimpulso: como la acción correctiva llega tarde, el sistema tiende a superar la posición deseada antes que el controlador pueda reaccionar.
- Respuesta oscilatoria y lenta: para compensar el retrado y evitar inestabilidad. hay que reducir la ganancia del controlador.
- Limita la ganancia del controlador.

El máximo tiempo de retardo medido fue de $L = 500\,seg$, y el lazo cerrado se comporta en ese caso:

He aquí la razón principal por la que se desea implementar el controlador con la

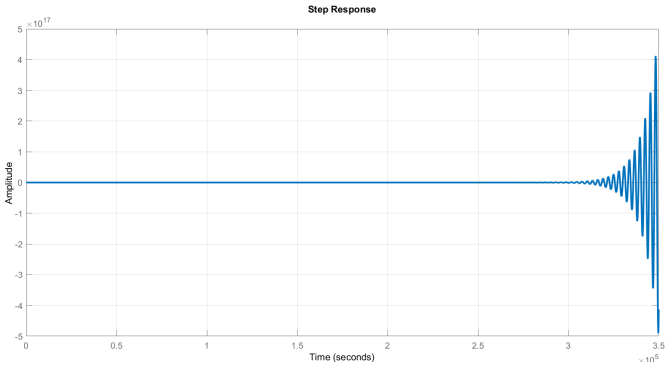


Figura 47: Lazo cerrado con controlador sintonizado con ZN con retardo $L = 800\,seg$

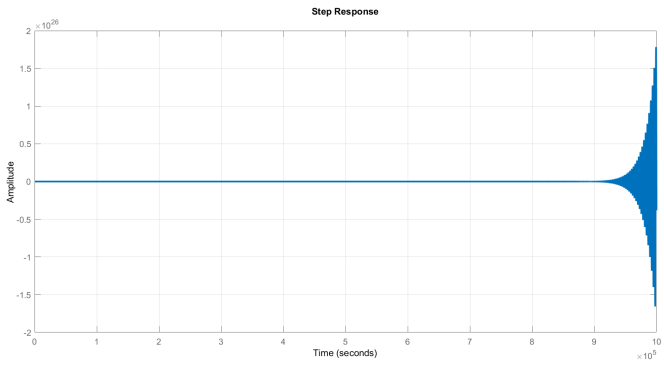


Figura 48: Lazo cerrado con controlador sintonizado con CC con retardo $L = 800\,seg$

sintonización por las reglas de Cohen Coon, y es porque (Figura 46), el sobreimpulso es menor (hasta en la condición más desfavorable), lo que quiere decir que se desvía menos respecto de la temperatura de referencia.

Con el retardo hay que tener cuidado, debido a que si en alguna caso es más grande todavía, se corre el riesgo de que se inestabilice el sistema, por ejemplo para $L = 800\,seg$

y en este caso, se debe reducir la ganancia del controlador implementado para poder evitar el comportamiento inestable.

6.2. Adición de anti-windup

El controlador que se obtiene es lineal, debido a que su diseño se basó en técnicas para sistemas lineales. El rango lineal del actuador (el difusor) se encuentra entre 20 % y 70 % de apertura, por debajo y por encima de este rango, se dice que el actuador satura, por lo que produce una no linealidad en el sistema obteniendo resultados no deseados.

Al contar con un actuador con limitaciones físicas (Figura 49) y una planta con un posible gran retardo, sería normal que ocurran estos problemas, ya que el controlador cuenta con una acción integral del error, que se va incrementando a medida que no percibe cambios en la salida. Este efecto se denomina **Windup**, y se produce porque el sistema tarda mucho en reaccionar antes un cambio en la acción de control, y para evitarlo o minimizarlo, se decide implementar una técnica conocida como **anti-windup**.

6.3. Control discreto

Una vez se obtienen los parámetros del controlador para lograr la dinámica deseada, se procede a discretizar el controlador calculado, y verificar el efecto al agregarlo al sistema. Para poder cumplir con este cometido, se utiliza el método de Tustin, el cual consiste en reemplazar:

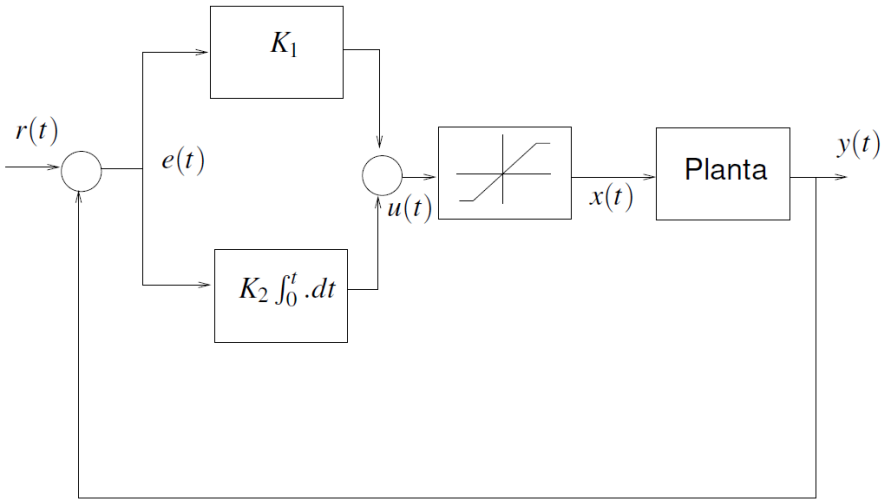


Figura 49: Sistema con saturación en el actuador y compensación PI

$$s \rightarrow \frac{2}{T_m} \times \frac{z-1}{z+1}$$

donde T_m es el período de muestreo. Para el controlador PI, al usar el método de Tustin (también conocida como la regla trapezoidal), se obtiene:

$$PI(z) = \frac{(T_m\,K_P\,+\,2\,T_i\,K_P)\,z + T_m\,K_P - 2\,T_i\,K_P}{2\,T_i\,z - 2\,T_i} \tag{49}$$

Lo próximo a realizar es llevar la funcón de transferencia en z a su ecuación de diferencias finitas equivalente. Para esto, se plantea la función de transferencia en función de su entrada y salida, y se expresa en términos de z^{-1}, es decir:

$$\frac{Y(z)}{X(z)} = \frac{(T_m\,K_P\,+\,+2\,T_i\,K_P) + (T_m\,K_P\,-2\,T_i\,K_P)\,z^{-1}}{2T_i\,-2\,T_i\,z^{-1}} \tag{50}$$

y resolviendo:

$$Y(z) = Y(z)\,z^{-1} + K_P\left(\frac{T_m}{2\,T_i}\,+\,1\right)X(z)\,+K_P\left(\frac{T_m}{2\,T_i} - 1\right)X_(z)\,z^{-1} \tag{51}$$

Y ya con esta última expresión, es posible pasar al dominio discreto:

$$y[n] = y[n-1] + K_P\left(\frac{T_m}{2\,T_i} + 1\right)x[n] + K_P\left(\frac{T_m}{2\,T_i} - 1\right)x[n-1] \tag{52}$$

Esta última es la ecuación de diferencias finitas que se implementa en el microcontrolador, y representa al controlador PI discretizado.

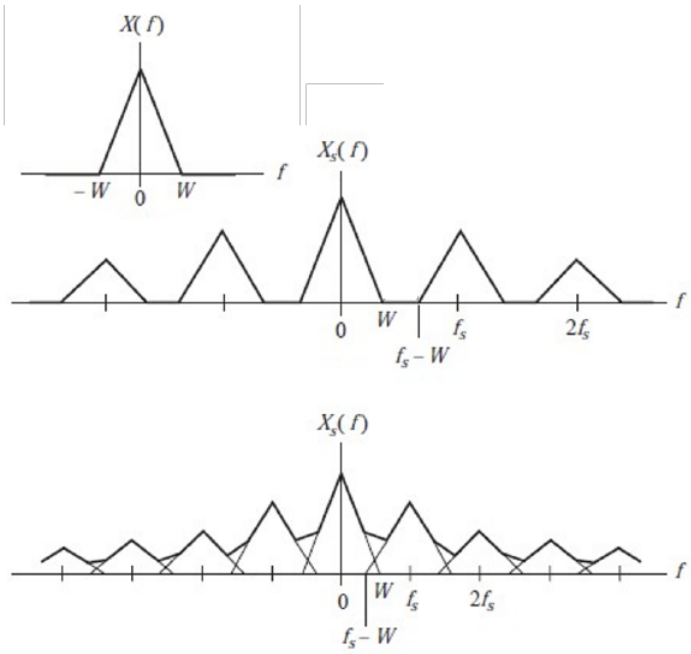


Figura 50: Muestreo de una señal de ancho de banda W

6.3.1. Frecuencia de muestreo

El muestreo de una señal causa, en el espectro de la señal muestreada, una superposición del espectro original que depende de la frecuencia de muestreo elegida. Para evitar la superposición espectral (o aliasing) se procede a muestrear a una frecuencia mayor o igual al doble del ancho de banda del espectro (frecuencia de Nyquist, Figura 50).

Como se mencionó en secciones anteriores, se tiene una componente de mayor frecuencia que la del polo dominante (dinámica de la planta), que es la que introduce el actuador, por lo que la frecuencia de muestreo seleccionada será más grande que ésta.

6.3.2. Respuesta obtenida con el controlador discreto

Si se convierte a dominio discreto el lazo de calculado, se obtiene una aparentemente igual a la que se obtiene en el dominio continuo, pero si se realiza un acercamiento a la señal, se aprecia que aparece **escalonada**, dónde cada permanencia de la salida en un valor dura lo que dura el tiempo de muestreo seleccionado (Figura 51).

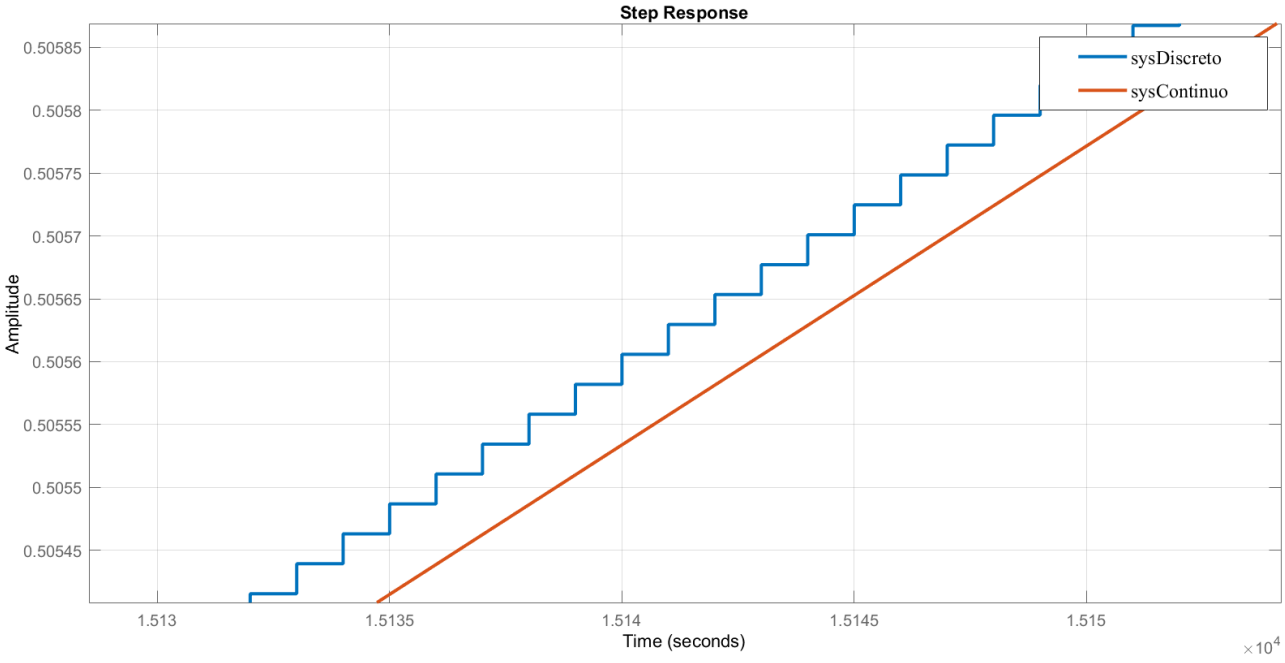


Figura 51: Comparación sistema continuo vs. discretizado

6.3.3. Ecuación de diferencias con anti-windup

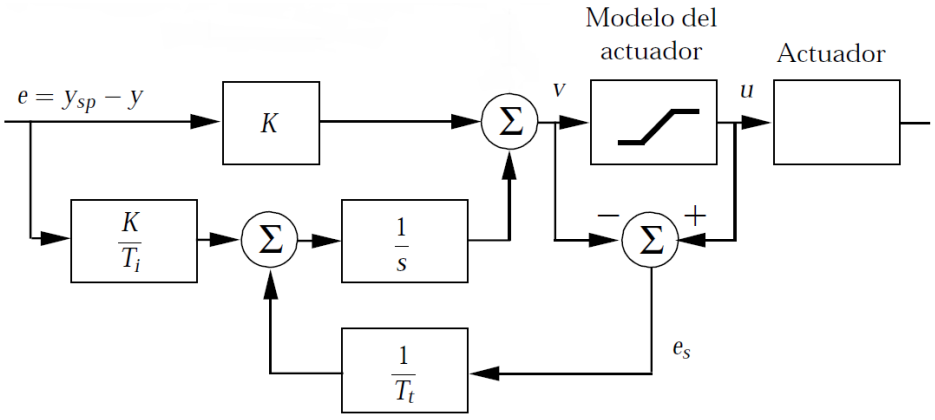


Figura 52: Diagrama en bloque con Anti-Windup

Con el fin de determinar la ecuación de diferencias, se debe tener en cuenta que la acción descargadora influye únicamente en la acción integradora (Figura 52), por lo que se llega a:

$$u^*(z) = K_p\, e_{(z)} + K_i\, \frac{1}{z-1}\, T_s\, (e_{(z)} + j_{(z)}) \tag{53}$$

7. Diseño, desarrollo y construcción del prototipo

En este capítulo se detalla el proceso de pensamiento llevado a cabo para la materialización de la solución propuesta. Se comienza planteando los requerimientos de diseño deseados, conceptos técnicos necesarios y la selección de componentes.

Se justifica la elección del material y software utilizado, para continuar con la presentación de las distintas étapas del diseño.

7.1. Conceptos y requerimientos de diseño

Los difusores del DIE que se observan en la Figura 22 poseen un sistema de apertura complicado de automatizar y al ser piezas metálicas se necesita de mayor potencia para poder moverlas, aún sin tener en cuenta la acumulación de polvo o las imperfecciones de los mismos que interfieran en el movimiento.

Se busca entonces, diseñar un nuevo difusor, que tenga una apertura regulable mediante mecanismos mecánicos como engranajes cuyo movimiento sea controlado por mecanismos electrónicos, como un servomotor o motor paso a paso. Se desea que sean del mismo tamaño que los difusores actuales para un fácil reemplazo de los mismos.

Se piensa en un diseño modular para facilitar las tareas de mantenimiento sin necesidad de desinstalar toda la estructura y que tenga la capacidad de procesamiento local necesaria para ejecutar el lazo de control mientras se conecta de forma inalámbrica, que permita recibir y transmitir datos de temperatura y estados.

7.1.1. Mecanismo de apertura de diafragma tipo iris

Como opción óptima para regular el grado de apertura del difusor el mecanismo que se eligió es inspirado por los de diafragma tipo iris, se puede observar uno en la Figura 53a. Este mecanismo permite controlar la temperatura del ambiente ajustando el flujo de aire en los conductos circulares.

(a) Difusor cerrado

(b) Mecanismo de apertura

Figura 53: Difusor de diafragma tipo iris comercial

En general el mecanismo de apertura de este tipo de difusores es dado por una palanca que se encuentra fuera del perímetro externo y cuyo movimiento, en sentido horario o antihorario, regula la apertura de las "aletas" internas.

7.1.2. Control del mecanismo de apertura mediante un servomotor

En un principio, se pensaba controlar el grado de apertura del difusor con el uso de un motor paso a paso. Se puede observar el esquemático de uno en la Figura 54.

Este tipo de de motores divide una rotación completa en un número específico de pasos discretos e iguales, por lo que si se quiere mover el motor a una posición deseada, el controlador envía una serie de pulsos eléctricos donde cada pulso provoca que el motor avance un paso. Operan sin realimentación, es decir que el controlador asume que el motor completó el movimiento que le pide, pero no verifica la posición real.

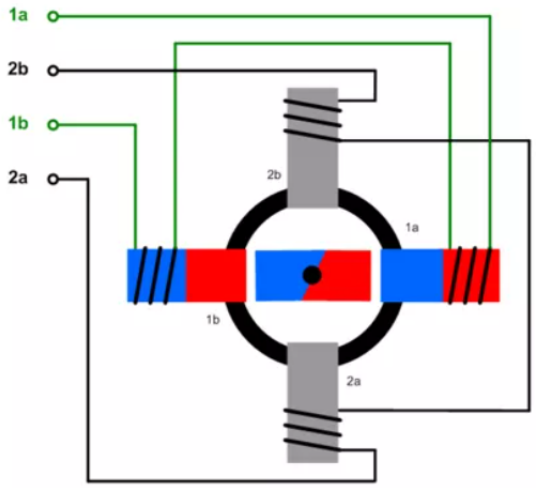


Figura 54: Esquema de un motor paso a paso

A diferencia de estos, un servomotor utiliza un sistema de retroalimentación para garantizar que la posición, velocidad y torque sean los deseados. Existe, entonces, una comunicación constante entre el controlador y el motor, permitiendo que el primero sepa la ubicación del segundo después de darle una orden de movimiento, conformando un sistema de lazo cerrado.

7.1.3. Engranajes

Lo innovador del diseño, está en el empleo de engranajes para la apertura y cierre de las aletas del difusor diafragma iris.

Los engranajes son ruedas dentadas que se acoplan entre sí con el fin de poder transmitir eficientemente movimiento y fuerza [14]. En la combinación de dos engranajes, se denomina motor al que aporta la fuerza de entrada y conducido al que se acopla para obtener una salida controlada.

En este caso el que mueve el servomotor es el engranaje motor y el engranaje conducido el que mueve las "aletas" del difusor. En la Figura 55 se puede observar un ejemplo de esto.

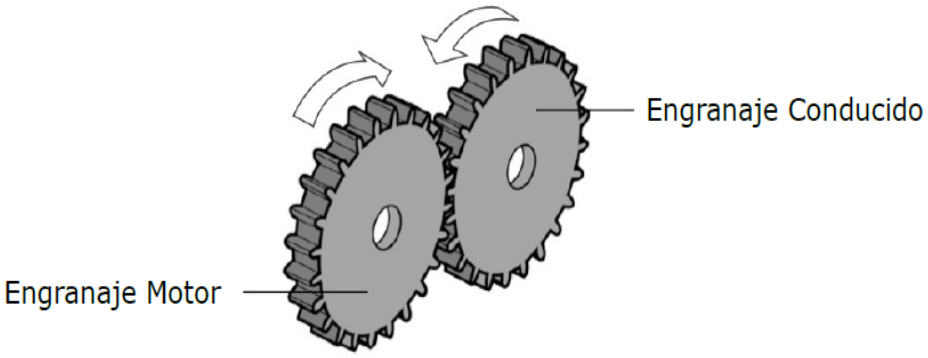


Figura 55: Ejemplo de engranaje motor y conducido

Al acoplar dos engranajes se obtienen cambios de dirección, velocidad y fuerza, pero siempre que se quiera incrementar alguna de estas variables, otra va a disminuir de forma proporcional.

En cuanto al **sentido de giro**, el acople de un engranaje motor (M) con otro conducido (C) genera en este último un giro opuesto al del primero.

Se logra hacer girar un engranaje conducido en el mismo sentido que el motor, añadiendo otro, denominado *loco*, entre ellos, pero no es el caso de estudio.

Se conoce como **relación de engrane o de transmisión (i)** al cociente entre el número de dientes del engranaje conducido (Z_C) y el número de dientes del engranaje motor (Z_M).

$$\frac{Z_C}{Z_M} = i \tag{54}$$

Esta relación constituye un importante paso para el diseño, para poder conocer bien cuantas vueltas del engranaje motor hacen falta para desplazar el conducido de forma tal que llegue a abrir por completo el difusor.

En cuanto a la velocidad de salida del eje conducido, esta se obtiene multiplicando la velocidad de entrada, la del eje motor, por el inverso de la relación de engrane.

$$\omega_C = \omega_M \times \frac{Z_M}{Z_C}$$

$$\omega_C = \frac{\omega_M}{i} \tag{55}$$

El momento de salida del eje conducido se obtiene multiplicando el momento de entrada del eje motor por la relación de engrane:

$$M_C = M_M \times \frac{Z_C}{Z_M} \rightarrow M_C = M_M \cdot i$$

$$M_C = M_M \times i \tag{56}$$

En el Anexo 12.2: *Fundamento de las relaciones entre engranajes*, se puede ver el fundamento detrás de estas relaciones entre los engranajes.

7.2. Hardware utilizado

En esta sección se presentan las soluciones comerciales utilizadas tanto para los reguladores y microcontroladores.

7.2.1. ESP32-C6-Zero

La **ESP32-C6-Zero** es una placa de desarrollo que cuenta con un microcontrolador de bajo costo y alto rendimiento. Su diseño compacto, su gran variedad de interfaces perifericas y la capacidad de conexión inalambrica Wi-Fi la vuelven la solución ideal para nuestro producto.

Además de las prestaciones, una de las principales ventajas que tiene es su fácil configuración y programación en distintos entornos de desarrollo. Para este proyecto se utilizó el IDE[4] de Arduino. Motivó la elección del entorno de programación la búsqueda de un equilibrio entre un código óptimo y un rápido desarrollo para implementar el prototipo.

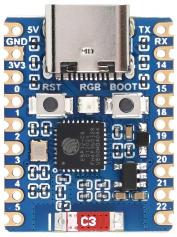


Figura 56: ESP32 C6 Zero

Esta placa se utiliza para implementar el algoritmo de control, cuyo código se encuentra en los anexos.

7.2.2. ESP8266 Modelo 01S

La ESP8266 modelo 01S, también llamada ESP01S a secas, es una placa de desarrollo compacta y de bajo costo que integra un microcontrolador con conectividad Wi-Fi, siendo por esto ampliamente utilizada en aplicaciones de IoT. Gracias a su

[4] *Integrated Development Environment* - Entorno de Desarrollo Integrado. Software que combina herramientas esenciales para la creación de programas en una sola interfaz.

simplicidad, tamaño reducido y compatibilidad con múltiples entornos de desarrollo, resulta adecuada para aplicaciones que requieren conectividad inalámbrica básica y bajo consumo de recursos. En la Figura 57 se puede ver el modelo exacto utilizado.

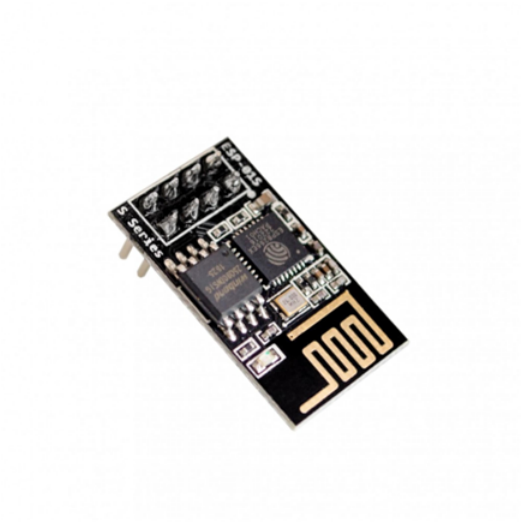


Figura 57: ESP8266-01S

Así como en el caso del microcontrolador encargado del control, se elige el entorno de desarrollo de Arduino IDE para programarla, facilitando así un inicio rápido y su integración en productos.

Esta placa se utiliza para sensar temperatura, y poder enviarla al broker Mosquitto, vía MQTT.

7.2.3. Modulo Regulador Switching Ajustable LM2596

El módulo regulador switching ajustable **LM2596** es un convertidor *DC-DC* tipo *buck* (reductor), ampliamente utilizado en aplicaciones electrónicas que requieren una regulación eficiente de tensión. Se basa en el circuito integrado LM2596, un regulador conmutado de alta eficiencia capaz de entregar corrientes de salida de hasta 3 A, lo que lo convierte en una solución adecuada para alimentar microcontroladores, sensores y otros dispositivos electrónicos desde fuentes de mayor tensión.

Este módulo permite convertir una tensión de entrada continua en un valor inferior ajustable mediante un **potenciómetro integrado**, lo que facilita su configuración sin necesidad de modificar componentes externos. Generalmente admite tensiones de entrada en un rango aproximado de 4 V a 40 V, mientras que la tensión de salida puede ajustarse desde alrededor de 1,25 V hasta valores cercanos a la tensión de entrada, manteniendo una regulación estable incluso ante variaciones de carga.

Una de las principales ventajas del LM2596 frente a los reguladores lineales es su **alta eficiencia**, que puede superar el 80 %–90 %, reduciendo significativamente la disipación de potencia y el calentamiento del circuito.

El módulo incorpora los componentes externos necesarios para el funcionamiento del regulador, tales como el inductor, el diodo Schottky y los capacitores de filtrado, lo que simplifica su integración en sistemas electrónicos. Se decide implementar para el proyecto esta solución comercial (Fig.58), ya que viene toda la placa armada con el regulador, lista para convertir tensión, y tiene el mismo precio que los reguladores LM-2596 discretos.

Figura 58: Convertidor LM2596

Se utilizan dos de estos convertidores, para alimentar la placa ESP8266 y el servomotor, que se encuentran ubicados en el techo junto con el difusor.

7.2.4. Interacción servomotor con el controlador

Figura 59: Servomotor

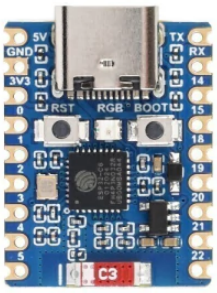


Figura 60: Controlador

Con el servomotor (Fig.59), y la ESP32-C6-Zero (Fig.60) se puede:

- **Orden del controlador:** el proceso comienza cuando el controlador implementado en el micro envía una señal al servomotor indicando una posición o velocidad deseada (esa señal es el **setpoint**).
- **Movimiento del servomotor:** el servomotor recibe esa señal y gira hasta alcanzar la posición deseada.

- **Monitoreo del encoder (señal de retroalimentación):** mientras el motor se mueve, el encoder monitorea continuamentre la posición y velocidad del eje real, y convierte el movimiento mecánico en una señal eléctrica que representa la posición exacta en ese momento.
- **Corrección en tiempo real:** la señal del encoder se envía de vuelta al controlador, y si es necesario se ajusta la señal para que el motor vaya a la posición deseada

Y este ciclo de realimentación se repite miles de veces por segundo.

7.2.5. Interacción del sensor de temperatura con el microcontrolador del sensado

El nodo sensor consiste en la **ESP8266-01S** que envía el dato de la temperatura actual mediante MQTT al broker, proveniendo este datode un sensor DHT22 (Fig.61) que registra permanentemente la temperatura actual de la sala.

Figura 61: Sensor DHT22

El sensor **DHT22** es un dispositivo digital utilizado para la medición de **temperatura** y **humedad relativa**. Integra un sensor capacitivo de humedad y un termistor, ofreciendo lecturas estables y calibradas digitalmente. Opera con una tensión de alimentación entre **3.3 V y 6 V** y se comunica mediante un **protocolo digital de un solo hilo**, lo que facilita su integración con microcontroladores. Presenta un rango de medición de temperatura de **-40 °C a 80 °C** con una precisión de **±0.5 °C**, y un rango de humedad relativa de **0 % a 100 %** con una precisión de hasta **±2 %**.

7.3. Impresión 3D

Para llevar a la realidad las piezas diseñadas, se optó por la tecnología de impresión 3D, también conocida como Manufactura Aditiva FDM, dejando de lado otros métodos más tradicionales, como el mecanizado CNC, aunque no se descarta en un futuro pasar a algún otro modelo de creación.

El mecanismo del difusor tipo iris pensado es una pieza única que requiere detalles específicos y personalizados, su fabricación con otros métodos habría llevado meses

entre pruebas, modificaciones y arreglos debido a los tiempos típicos de trabajo de los talleres. La impresión 3D permitió reducir el ciclo de validación de las distintas piezas, lo que permitió corregir el diseño y volver a fabricar las piezas en el mismo día.

7.4. Selección de Material: Filamento PETG

El filamento elegido para las impresiones 3D fue el **PETG**[5], esto debido a las condiciones ambientales de operación del dispositivo. Puede verse el mismo en la Figura 62.

El difusor estará conectado directamente a las salidas del sistema de calefacción ubicadas en el techo, por lo que estará sometido a flujos de aire caliente constantes y debe soportar el peso de los ductos. Allí es donde el filamento elegido cobra importancia, mientras el conocido PLA[6] tiene una *temperatura vítrea*[7] (T_g) de aproximadamente 60°C el PETG mantiene sus propiedades mecánicas hasta unos 80°C.

Figura 62: Filamento plástico

Por otra parte, el PETG posee más resistencia ante los impactos y mayor flexibilidad, lo que fue aprovechado en el diseño de las uniones mecánicas por encastre, evitando así los riesgos de fractura.

7.5. Software de Diseño - Fusion 360

El diseño del nuevo difusor, que reemplazará los actuales, se llevó a cabo en el software de diseño 3D **Autodesk Fusion**, anteriormente conocido como *Fusion 360*. Como su nombre lo indica, es uno de los tantos otros programas creados por la compañía *Autodesk*, marca reconocida ya por AutoCAD, Inventor y Tinkercad, entre otros.

En la Figura 63, se puede observar el logo del programa y se accede al mismo a través del siguiente enlace: https://www.autodesk.com/latam/products/fusion-360.

[5] PoliEtileno Tereftalato Glicol, un termoplástico conocido por su resistencia térmica y mecánica.

[6] Ácido Poliláctico: polímero biodegradable derivado de recursos renovables. Es el estándar en impresión 3D por su facilidad de uso.

[7] Temperatura a la cual el material pasa de ser duro a blando.

Figura 63: Software Autodesk Fusion

Se eligió Autodesk Fusion ya que es una herramienta potente con las siguientes características:

- **Diseño 3D:** Permite el modelado de piezas 3D de forma libre.
- **Fabricación (CAM) e Ingeniería (CAE):** Ofrece herramientas para analizar la forma, el ajuste y la estética de los productos antes de la fabricación.
- **Exportación para impresión 3D:** Permite la exportación de las piezas a distintos formatos de impresión 3D reconocibles por softwares de impresión 3D, permitiendo elegir la cantidad de detalles y polígonos deseados para el modelo 3D.
- **Procesamiento en la nube:** A diferencia de otros software de diseño que exigen que el usuario tenga una computadora de alto rendimiento, el procesamiento en la nube de Autodesk permite modelar piezas complejas utilizando hardware de menor nivel, como lo son las notebooks de los miembros del equipo.
- **Colaboración en la nube y control en la nube:** La plataforma centraliza el almacenamiento de los archivos de trabajo en la nube, permitiendo compartirlos y guardar un historial de versiones. Esto permite que el equipo trabaje sobre la última versión estable del modelo, evitando conflictos de archivos duplicados o desactualizados y, ante cualquier inconveniente, poder volver a una versión anterior. Al tratarse de un proyecto grupal, la correcta gestión de archivos resulta fundamental.
- **Estudio del movimiento:** Permite hacer una validaciòn virtual a través de las herramientas de ensamblaje y animación, permitiendo simular el comportamiento mecánico de las piezas. Permitió verificar la apertura y cierre de las aletas y detectar las colisiones entre los engranajes, ahorrando así tiempo y material en prototipos que habrían fallado.

7.5.1. Licencia educativa

Autodesk ofrece una versión gratuita del software para uso personal; sin embargo, esta posee limitaciones críticas, ya que restringe la exportación de archivos para su manufactura. A su vez, Autodesk ofrece la posibilidad de reclamar una *licencia educativa* para estudiantes, la cual se reclamó y se obtuvo sin inconvenientes al ser estudiantes universitarios de una institución de educación pública, como lo es la Universidad Nacional de la Patagonia San Juan Bosco. Gracias a la acreditación institucional, se desbloqueó la capacidad de exportar los modelos en formatos de alta calidad (STL/STEP) necesarios para la etapa de impresión 3D.

7.6. Evolución del prototipo

El proceso de diseño se estructuró como un ciclo iterativo de validación incremental. Esta metodología permitió desglosar la complejidad mecánica en etapas manejables, comenzando por la validación de modelos existentes hasta llegar a la ingeniería de detalle de una solución personalizada.

De esta manera, se ajustaron los parámetros de diseño y la selección de componentes electromecánicos antes de comprometer recursos en la versión final.

7.6.1. Primera etapa: Investigación

En primera instancia se realizó una investigación de diseños ya existentes en la plataforma *Thingiverse*[8]. Se optó por el modelo llamado "*Servo automated iris / aperture for air flow controll*" , disponible en el siguiente enlace: www.thingiverse.com/thing:3563742. Se lo puede observar en la Figura 64.

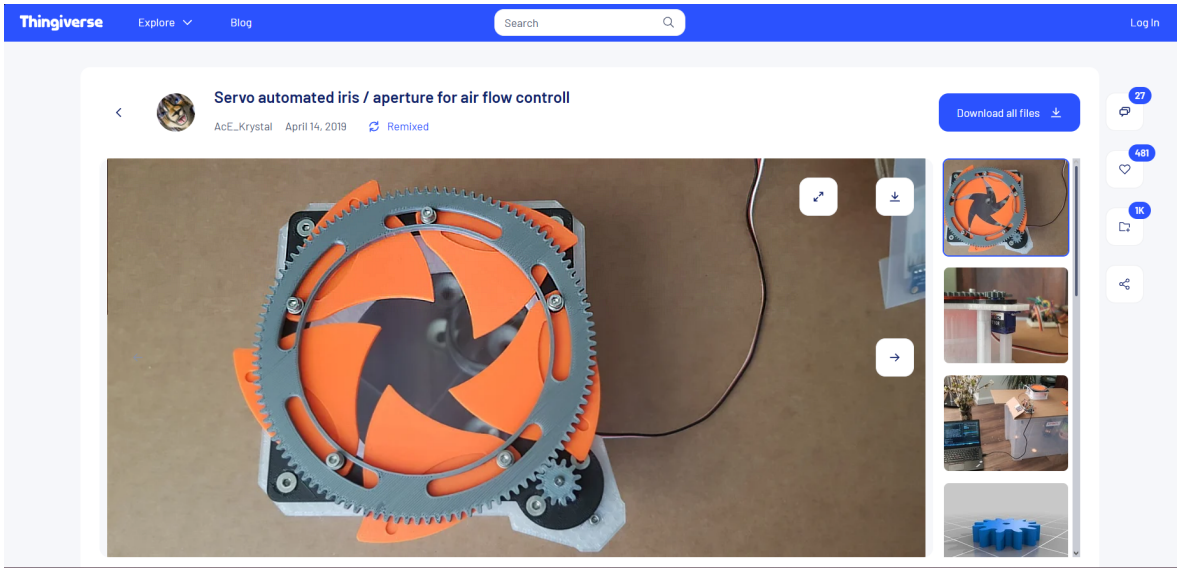
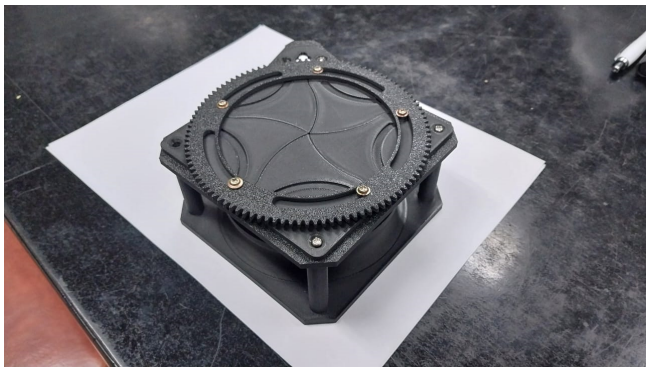


Figura 64: Modelo de referencia utilizado para la comprensión inicial del mecanismo

Una vez impreso, ver Figura 65, se estudiaron los mecanismos de diafragma de iris con el objetivo de comprender la relación de movimiento entre el anillo dentado y el desplazamiento radial de las aletas. Para la impresión de este prototipo se utilizó filamento PLA ya que no sería sometido a las condiciones de trabajo.

[8]Comunidad de repositorios en línea gratuitos donde diseñadores y aficionados comparten, descargan y crean modelos 3D listos para ser impresos.



(a) Vista en perspectiva del mecanismo iris impreso cerrado

(b) Vista superior del mecanismo iris impreso abierto

Figura 65: Impresión del modelo 3D obtenido en Thingiverse.

Este diseño estaba dimensionado para el servomotor SG90[9], el cual se puede observar en la Figura 66a. El mismo carece de la fuerza necesaria para mover las aletas del difusor a tamaño real. Además, el modelo descargado solamente contiene el modelo de malla para la impresión, el cual se puede escalar, pero eso implica aumentar proporcionalmente las dimensiones de todos los detalles, tales como las perforaciones para tornillos, encastre para el servomotor, los dientes de los engranajes, etc.

(a) Servomotor SG90

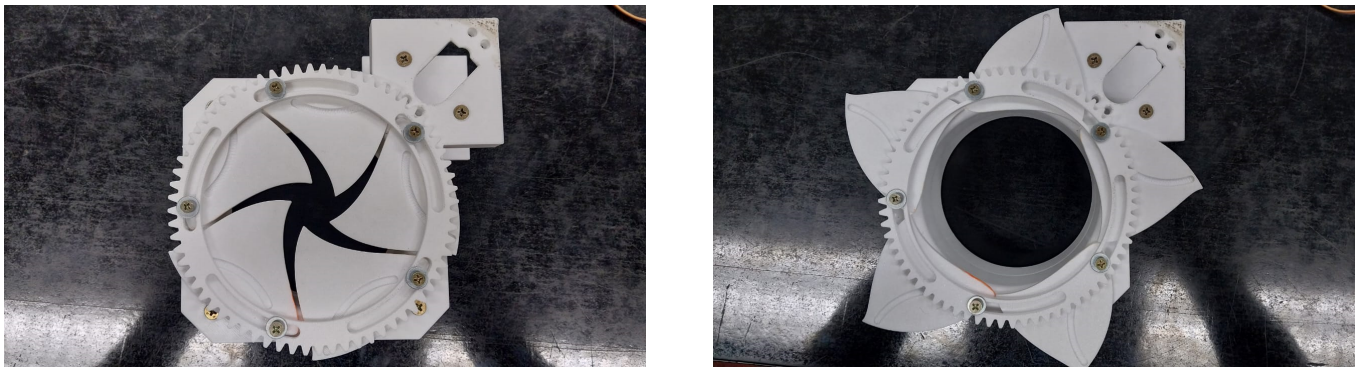
(b) Servomotor MG996R

Figura 66: Servomotores utilizados en el proyecto.

7.7. Segunda etapa: Rediseño

A partir del modelo impreso en la primera etapa y con las observaciones realizadas se creó este primer modelo "beta", el cual se puede observar en la Figura 67.

[9] Un micro-servomotor económico y popular, ideal para proyectos de robótica de bajo torque.



(a) Vista superior del modelo “beta” cerrado (b) Vista superior del modelo “beta” abierto

Figura 67: Impresión del primer diseño “beta”.

Se realizaron las mediciones con un calibre y se tomó la decisión de hacer un diseño más robusto, con piezas de mayor espesor y más resistentes. Se rediseñó el alojamiento del servomotor, para uno modelo MG996R[10], que se puede observar en la Figura 66b. El mismo es capaz de garantizar una apertura y cierre fluidos del difusor.

El modelo fue hecho a la misma escala que la versión anterior, para ahorrar material y verificar el correcto funcionamiento de los mecanismos.

En la Figura 68 se puede observar el modelado 3D hecho en fusion desde una perspectiva que permite ver las bases tanto superior como inferior y los pilares que unen ambas.

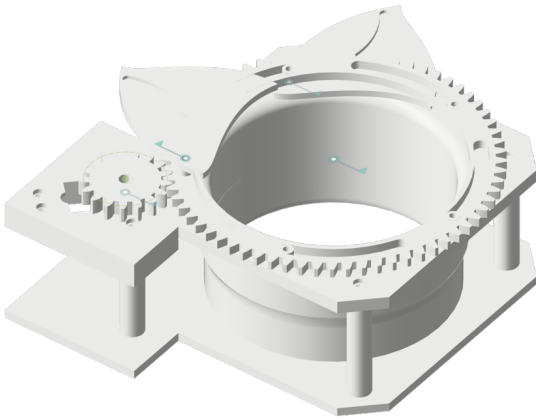


Figura 68: Modelo 3D del primer prototipo

7.7.1. Inconvenientes

Surgieron ciertas complicaciones con esta pieza que ayudaron a comprender el trabajo con impresiones 3D y su modelado. El primer inconveniente fue la cara de impresión seleccionada para la tapa inferior del difusor, los cilindros de la misma donde

[10]Servomotor digital popular y robusto. Ofrece una rotación de 180°

se colocan los tornillos dejan al resto de la estructura en el aire, por lo que es necesario utilizar muchos soportes para la impresión de la pieza, gastando material y dejando una superficie rugosa.

El segundo inconveniente fue que, debido a un error de cálculo, el soporte del servomotor se encontraba más alto, se corrigió así para la nueva versión. El tercer y último inconveniente fue que el engranaje que mueve el servomotor colisionaba con las aletas al abrirse, por lo que se lo limó en esta versión y se arregló para la próxima.

7.8. Diseño Final: Primer acercamiento

En esta versión se busca solventar los errores comentados anteriormente y llevarlo a la escala del producto final. Se lo trata en dos partes, el cuerpo principal que corresponde al difusor en sí y el módulo o "caja" que alojará la electrónica del proyecto de la forma más compacta posible.

7.8.1. Cuerpo principal

Para está parte del producto que es la que se encontrará en contacto con los ductos de ventilación se busca librarse de las bases tanto superior como inferior y los pilares que las unen. Se llega así a la idea de que el mismo cilindro del cuerpo tenga una rosca exterior a lo largo de toda su extensión y con una tuerca del mismo tamaño confinar el cielorraso entre la base del cilindro y la tuerca, independizándose así de tener que utilizar herramientas como destornilladores para su colocación.

En la Figura 69a se observa el modelo 3D del cuerpo principal desde una perspectiva que permite apreciar la rosca exterior que lleva el cilindro principal, la tuerca del mismo tamaño y las aletas cerradas. En la Figura 69b al cuerpo principal del difusor abierto, con las aletas separadas.

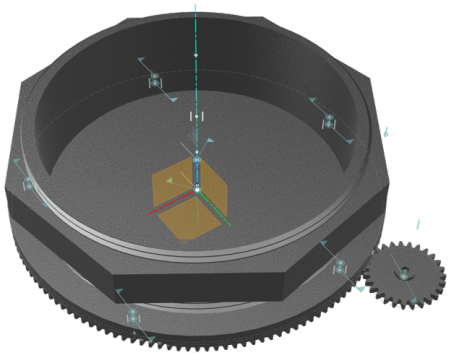
En la Figura 69c se observa el modelo 3D completo, donde el módulo auxiliar que posee la parte electrónica se acopla al cuerpo principal.

7.8.2. Módulo auxiliar

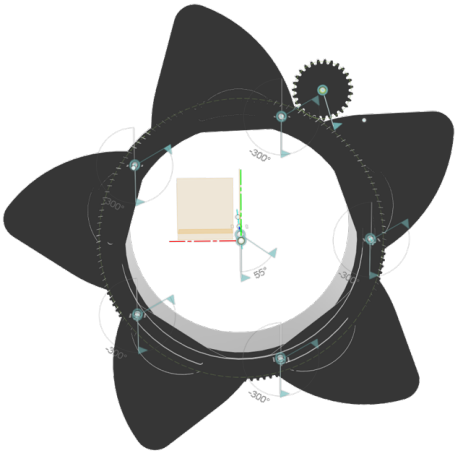
Este módulo fue pensado para alojar además del servomotor encargado de mover los engranajes, la placa de desarrollo ***ESP32-C6-Zero***, las dos placas comerciales de circuitos reguladores de tensión que utilizan al regulador LM2596 y el cooler.

En las Figuras 70a y 70b se observa la distribución y ubicación de los componentes electrónicos, además de la protuberancia que une ambas piezas, los "tornillos" y roscas que permiten la fijación de esta caja al cielorraso, siguiendo la misma lógica explicada para la sujeción del cuerpo principal. Se opta por paredes con huecos para que exista un flujo de aire que permita evitar que los reguladores levanten más temperatura de la normal al estar confinados, este flujo de aire es ayudado por el cooler/ventilador que se monta sobre la tapa del módulo.

Este prototipo llevó a conocer otros aspectos de las impresiones 3D, como considerar también la orientación de impresión para soportar distintos esfuerzos mecánicos.



(a) Perspectiva del modelo.

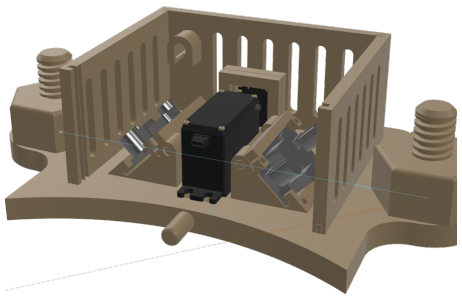


(b) Vista inferior del modelo.

(c) Cuerpo principal con el módulo que posee la parte electrónica

Figura 69: Modelo 3D del cuerpo principal.

(a) Vista de trasera de la caja



(b) Vista de frente de la caja

Figura 70: Primer diseño del contenedor de las partes electrónicas.

En la Figura 71 se puede observar una representación de lo ocurrido, donde al estar enroscando las tuercas, estas se clavaron al tornillo, debido a una mala terminación de la impresión, y ejercieron una torsión que quebró la base.

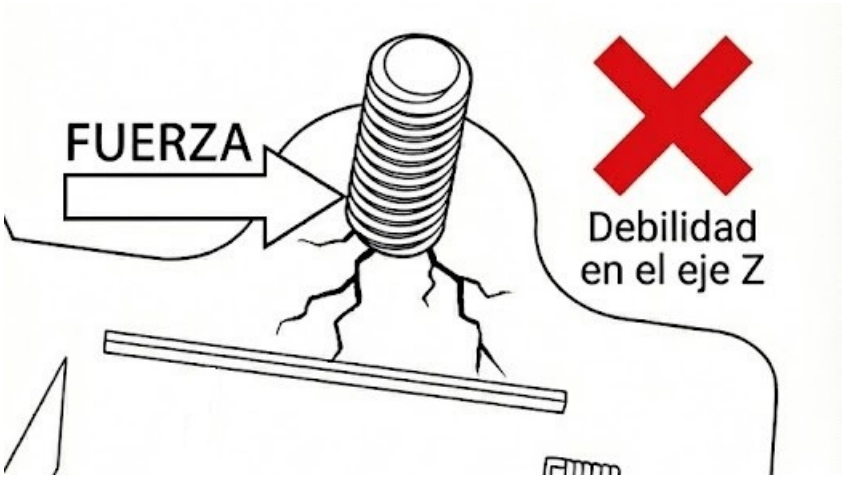


Figura 71: Representación gráfica de la debilidad estructural

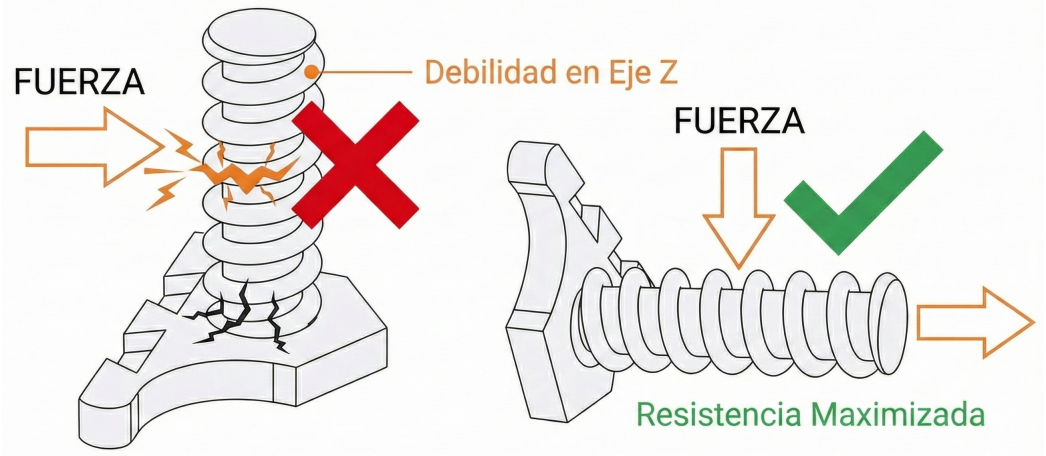


Figura 72: Representación gráfica de la distribución del esfuerzo mecánico.

7.9. Diseño final: Prototipo final

Para el prototipo final el cuerpo principal se mantuvo tal cual, se le hicieron modificaciones al módulo de la parte electrónica.

La modificación principal fue utilizar el tipo de rosca autorroscante que tiene mayor separación entre dientes, imprimiendola acostada para que cualquier esfuerzo mecánico aplicado se distribuya a lo largo de toda la pieza. Esto se encuentra ilustrado en la Figura 72.

Por otro lado, se da más espacio a los componentes para evitar cables ajustados y posibles cortes como así también mayor espesor a las paredes y a los ejes donde se deslizan para evitar quiebres. Todo esto lleva al diseño final que se observa en la Figura 73.

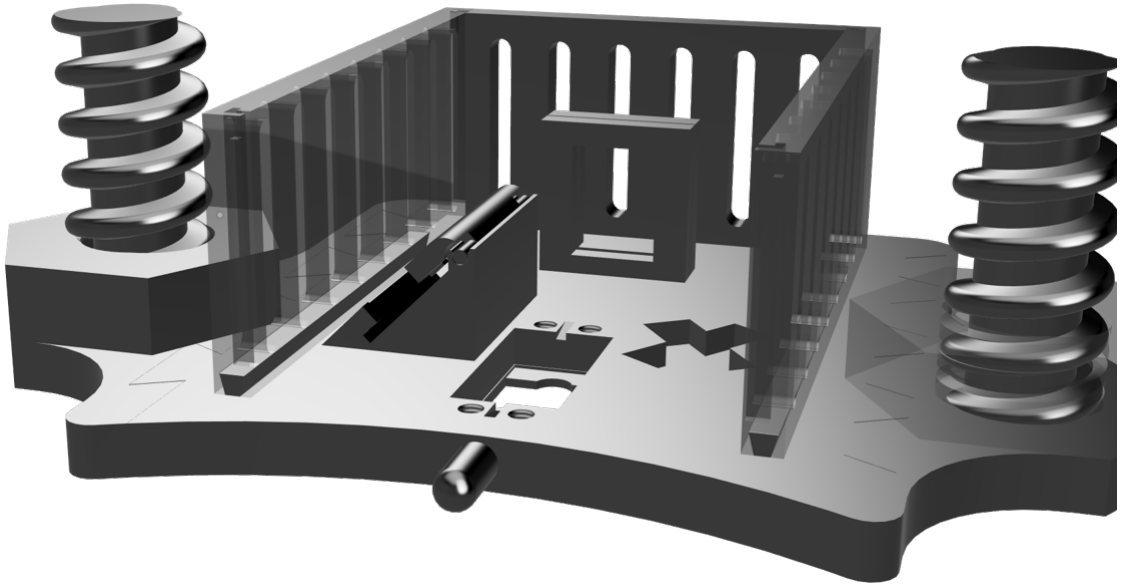


Figura 73: Diseño Final del módulo contenedor de la electrónica

7.10. Diseño modular

Como se ha visto a lo largo de esta sección, se ha optado por un diseño modular para el difusor, esta decisión se corresponde a limitaciones técnicas, como el tamaño de la cama de impresión de la impresora 3D, y a una estrategia de optimización de recursos y mantenimiento del producto final.

En la Figura 73 se puede ver como las paredes se montan sobre unos rieles y encastres sobre su parte superior para limitar el movimiento de las mismas.

7.10.1. Segmentación de la base

Debido a las dimensiones finales de la base, se decidió partir la misma en tres partes, siendo el centro donde se colocan los componentes y las partes izquierdas y derechas las de los tornillos para sujeción al techo. Resulta indispensable por el nuevo tamaño de la pieza y para poder imprimir con distintas orientaciones cada parte. Se pueden observar las tres partes en la Figura 74.

Hacerlo de esta manera lleva también a otras ventajas como mitigar los riesgos de manufactura. Si se imprimiera en su totalidad esto llevaría a tiempos elevados de impresión, donde si ocurre algún fallo, como atasco, corte de energía u otros, la pérdida de de material y tiempo sería total.

Para la unión de las tres secciones se diseño un sistema en encastre tipo “rompe-cabezas” donde para lograr una unión firme se trabaja con una tolerancia negativa de $-0{,}2$ mm en las interfaces de unión.

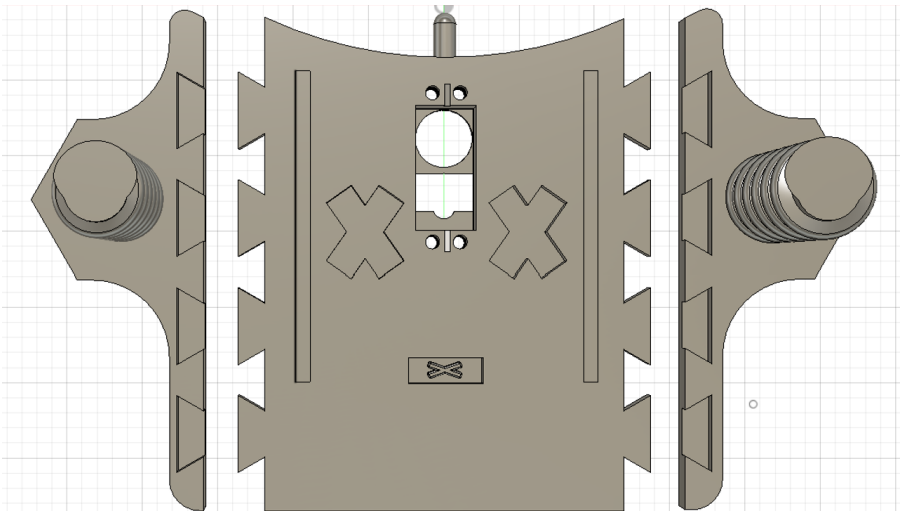


Figura 74: Despiece de la base del módulo.

7.10.2. Portaplacas

Con el objetivo de facilitar las tareas de mantenimiento y evitar el desgaste de la pieza plástica por el uso de tornillos, se diseñó un sistema al que se llamó "portaplacas". Existen dos tipos de portaplacas en la pieza, uno para la ESP32 y otro para los módulos comerciales de reguladores ajustables LM2596. Ambos pueden verse en la Figura 75.

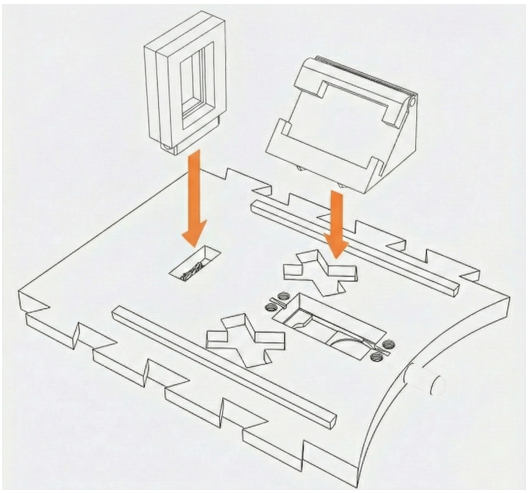


Figura 75: Ilustración del montaje de los portaplacas.

Estos portaplacas se diseñaron para encajar a presión en la base, lo que permite retirarlos y reemplazar las placas de forma cómoda en caso de averías. Si en un futuro se decide cambiar los componentes electrónicos, se puede diseñar un nuevo portaplacas para ese nuevo equipamiento, pero con el mismo encastre y así seguir utilizando la misma base.

7.11. Electrónica del diseño

7.11.1. Nodo Sensor

Este nodo es el que se encarga de tomar los datos de temperatura de la habitación, y consta de un sensor DHT11 conectado a una ESP8266, el cual envía los datos al broker, como muestra la siguiente figura:

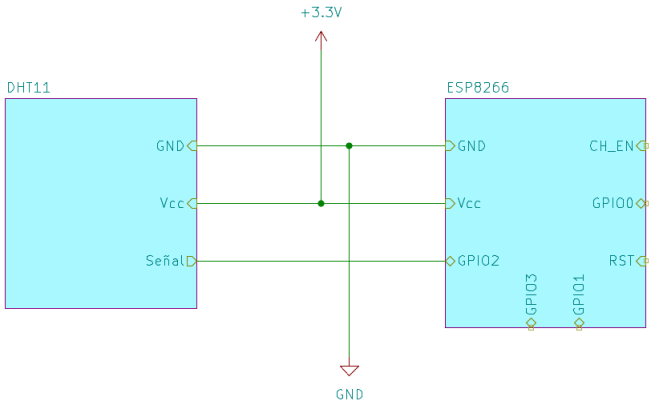


Figura 76: Esquemático de Sensor

En la implementación actual el sensor se conectó a la Raspberry Pi 3.

7.11.2. Nodo Actuador

Este nodo es el encargado aplicar la acción de control sobre el sistema, consta de dos reguladores de tensión a 5V, un servomotor, una ESP32-C6 y un cooler que refrigera el sitio donde se sitúa todo es te hardware, como se ve en la siguiente figura:

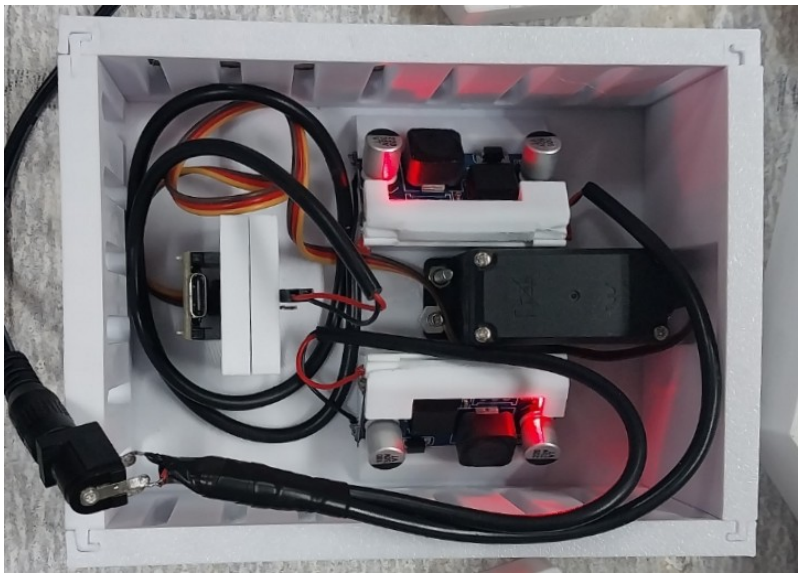


Figura 77: Electrónica de actuador implementada

En la figura 78 se puede ver el esquema correspondiente al nodo, nótese que se utilizan dos reguladores, esto es para poder separar la carga del controlador y así mantener una tensión estable en caso de que el motor se fuerce de más.

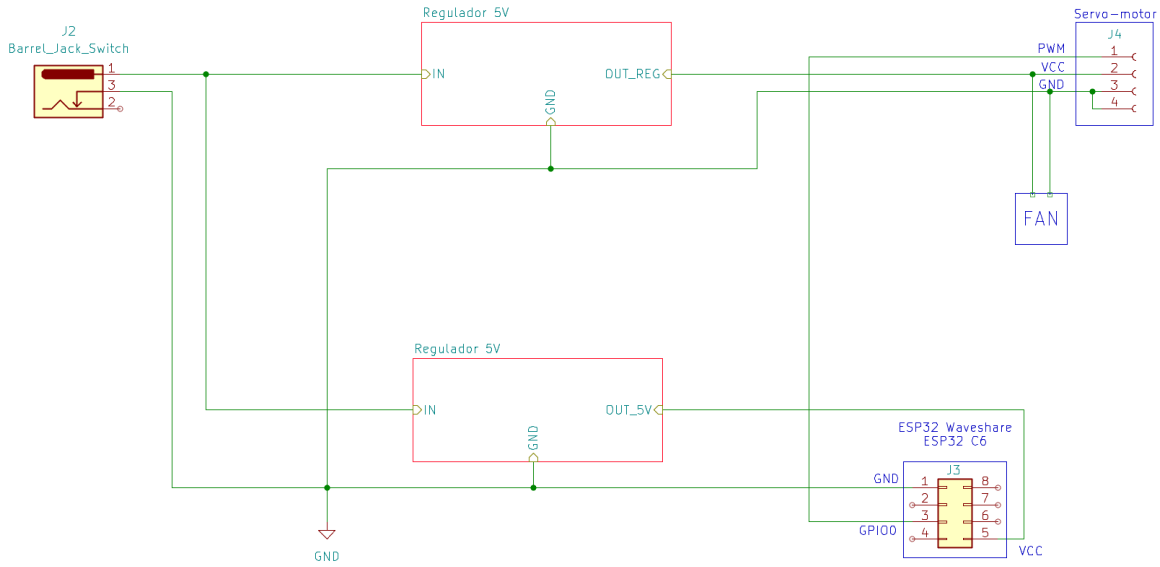


Figura 78: Esquemático de actuador

7.12. Impresión final

El modelo llevado a la realidad es el que se expuso en la sección anterior. Se puede observar su implementación en las Figuras 79 y 80.

Figura 79: Vista superior del producto montado sobre la placa de cieloraso.

En la Figura 79 se observa la parte del producto que no será vista por los usuarios finales, la que esta por arriba de la placa del cielorraso.

Figura 80: Vista superior del producto montado sobre la placa de cieloraso.

En la Figura 80 se observa la parte del producto que será vista por los usuarios finales, la que esta por debajo de la placa del cielorraso.

7.13. Implementación final

La implementación final se puede observar en la Fig.81, en dónde se puede ver que se encuentran las etapas de sensado (dónde está el sensor y el micro que sube la temperatura actual), el sistema central (que es la Rasberry con el broker Eclipse Mosquitto instalado allí) y microcontrolador del control (que será la ESP32 copn el algoritmo de control programado), quién recibe los datos desde la etapa del sensado y la temperatura de referencia (desde el Home Assistant).

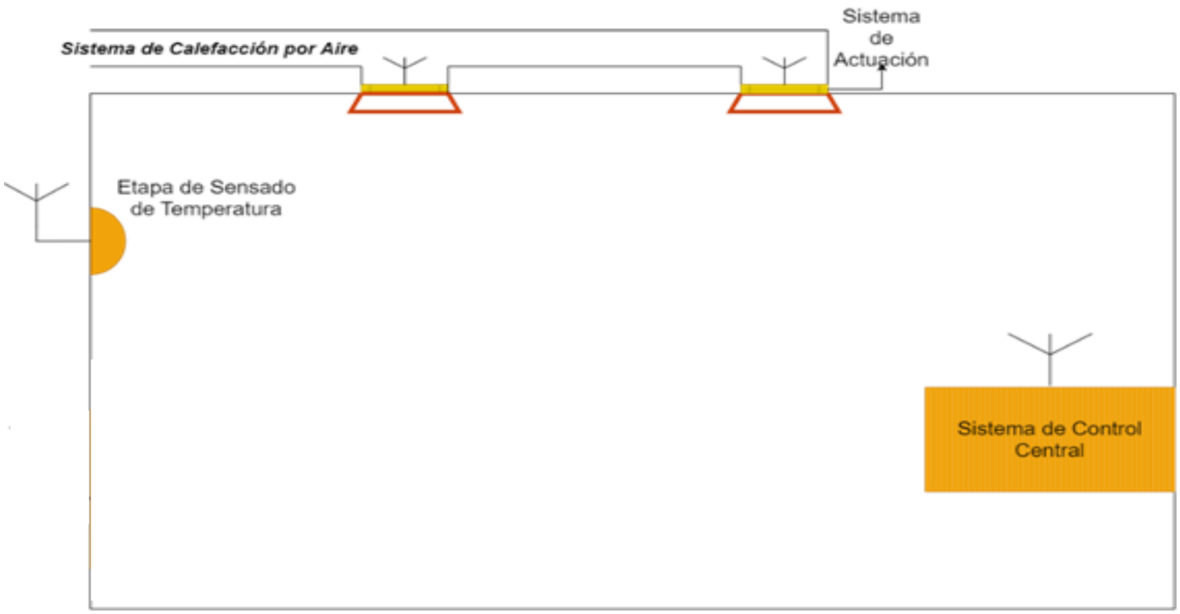


Figura 81: Sistema en la habitación

En la Fig.82 se observa en detalle cada una de las etapas y cómo es el vínculo entre ellas, hasta la etapa del actuador, que es el difusor, quien recibe la señal de control en caso de que se aleje la temperatura actual de la temperatura deseada.

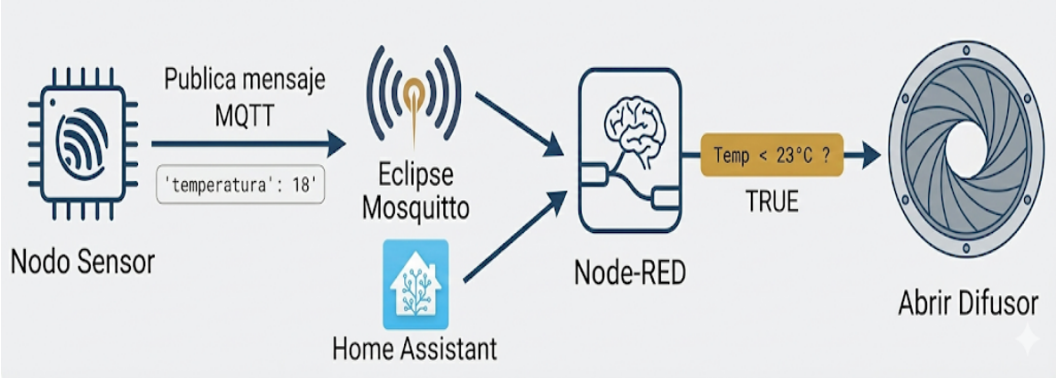


Figura 82: Flujo de diseño

8. Arquitectura de Software

En este capítulo se describe la arquitectura de software del sistema, se presentan los distintos programas (o *softwares*) utilizados y su implementación en el proyecto. La idea principal es un sistema de control central conformado por una *Raspberry Pi 3*, donde se instalan todos los servicios necesarios, permite la conexión de otros equipos a través de la red *Wi-Fi* y se puede monitorear de forma remota.

8.1. Central de datos: Raspberry Pi

La Raspberry Pi es una computadora pequeña y económica que presenta conectividad por Ethernet y algunos modelos por el estándar **Wi-Fi**.

El modelo que se usa para el proyecto será la Raspberry Pi 3, una como la que se encuentra en la Figura 83. La misma posee un CPU de 4 núcleos con arquitectura ARMv8 de 64 bits que corre a $1,4$ GHz y 1 GB de RAM. Soporta conectividad USB, HDMI, Ethernet, Wi-Fi de 2,4 GHz y de 5 GHz, y 40 pines configurables como entradas o salidas de propósito general (GPIOs).

Figura 83: Modelo utilizado: Raspberry Pi 3

El sistema operativo que utiliza es el **Raspberry Pi OS**, basado en Debian GNU/Linux, que está específicamente diseñado para la placa. En él se instalan y se ejecutan las aplicaciones para la transmisión, el almacenamiento y procesamiento de datos. Estas son:

- Node-Red;
- MySQL;
- Mosquitto;
- Grafana Server.

Su propósito en el proyecto es funcionar como **central de datos** o "host" del sistema. La elección de este *hardware* permite que el sistema sea modular, económico y fácil de reemplazar en caso de fallas físicas.

8.2. Presentación de las herramientas y programas utilizados

8.2.1. Node-RED

Node-RED es una herramienta, de código abierto, de *programación visual* basada en flujos, desarrollada originalmente por IBM para trabajar con IoT. Se ejecuta en el navegador y permite unificar dispositivos, APIs y servicios en línea.

También permite crear distintos tipos de funciones (generalmente en *JavaScript*, pero también permite ejecutar comandos del sistema), que permiten trabajar con los datos, como las que se encuentran en la Figura 84.

Una vez instalado en la central, se configura como servicio en el arranque del sistema. La programación basada en flujo define aplicaciones como "cajas negras" o nodos, que intercambian datos por conexiones preestablecidas que intercambian mensajes.

En este proyecto, Node-RED actúa como el **orquestador principal**:

- Recibe los datos crudos desde Mosquitto.
- Procesa y formatea la información.
- Decide cuándo guardar datos en la base de datos.
- Envía alertas o comandos a los controladores si es necesario.

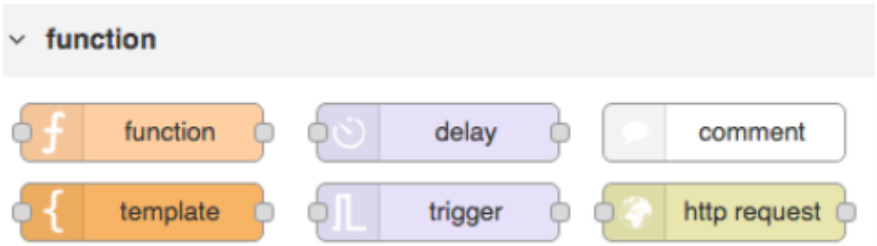


Figura 84: Ejemplo de nodos de proceso en el entorno de Node-RED.

8.2.2. Mosquitto

Mosquitto es un broker de mensajes de código abierto que implementa el protocolo MQTT[11]. Es un servidor liviano y de código abierto desarrollado por la fundación ECLIPSE, fundada en 2004 como una corporación sin fines de lucro.

Este software gestiona el intercambio de mensajes entre los que publican y los que se suscriben a un tema específico. Los clientes en este caso son los microcontroladores que miden la temperatura, los que ejecutan la acción de control y los demás programas. Es el corazón de las comunicaciones del sistema.

[11] Message Queuing Telemetry Transport

Se instala mosquitto utilizando el gestor de paquetes de la Raspberry, se lo configura para que se ejecute en segundo plano como un servicio del arranque del sistema.

8.2.3. Home Assistant

Es un sistema open source de domótica que permite controlar, automatizar y conectar dispositivos inteligentes. Se caracteriza por tener como objetivo garantizar la privacidad del usuario y por tanto no depende de suscripciones o **clouds** de terceras empresas, si no que funciona localmente en la propia red doméstica en la que se instala, sin riesgo de filtraciones de información y ofreciendo un control total sobre nuestros datos y el funcionamiento de nuestros dispositivos.

Las principales características de este sistema:

- Existe una capacidad total de personalizar las automatizaciones.
- Se pueden personalizar los dashboards para ver únicamente la información útil y necesaria en cada momento.
- Puedes crear tus propias integraciones para ampliar las funcionalidades de Home Assistant.
- Esta diseñado específicamente con la privacidad como objetivo, almacenando toda la información en local sin riesgo de fugas.
- Dispone de su propia aplicación móvil para interactuar con el sistema.
- Tiene un sistema de gestión de la energía predeterminado.
- Es compatible con cualquier pantalla para mostrar los dashboards.

Se utiliza entonces en el proyecto para permitir la interacción del usuario final mediante un *dashboard*[12]. Puede entonces elegir la temperatura deseada y ver la temperatura actual de la habitación desde su computadora o teléfono.

8.2.4. Grafana server

Es una plataforma de código abierto que tiene el fin de visualizar, analizar y monitorear datos, permitiendo realizar distintos tipos de gráficos (históricos, gráficos de barras) con consultas sencillas, con intervalo de tiempo seleccionable y configurable; de una gran variedad de bases de datos. La interfaz de usuario es intuitiva y permite ver valores históricos de forma sencilla.

8.2.5. Servidor Web y Proxy Inverso: Nginx

Nginx es un servidor web de alto rendimiento y código abierto. En este proyecto no se utiliza servidor web sino que cumple la función de **Proxy Inverso**. Su rol es recibir las peticiones HTTP externas y redirigirlas internamente al contenedor correspondiente (Node-RED o Home Assistant) según la dirección solicitada. Esto aporta seguridad

[12]Herramienta visual que organiza y presenta datos clave.

y comodidad, permitiendo acceder a los servicios mediante nombres de dominio locales (ej. `fedami.local`) en lugar de recordar direcciones IP y puertos específicos (ej. `192.168.1.15:1880`).

8.2.6. Control de Versiones: Git y GitHub

Para la gestión del código fuente se utiliza **Git**, el sistema de control de versiones distribuido estándar en la industria. Como repositorio remoto se emplea **GitHub**, una plataforma en la nube que aloja el código del proyecto. Esto permite:

- **Trazabilidad:** Mantener un historial de cada cambio realizado en el código.
- **Seguridad:** Si la tarjeta SD de la Raspberry Pi se daña, el código está respaldado en la nube.
- **Despliegue:** Facilita la actualización del sistema en producción mediante la descarga de la última versión estable.

8.2.7. Docker

Es una plataforma de código abierto para desarrollar aplicaciones en un *sandbox*[13]. Aunque los contenedores[14] existen desde 1979, Docker los ha hecho más accesibles. Con Docker, los desarrolladores pueden construir, probar y desplegar sus aplicaciones localmente o en un servidor de producción. En la Figura 85 se observa su distintivo logo de una ballena transportando contenedores.

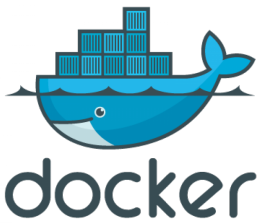


Figura 85: Docker

Desde el lanzamiento de Docker 1.0 en 2014, se ha estandarizado el uso de contenedores tanto para desarrolladores individuales como para empresas.

En Docker, se puede empaquetar una aplicación en una unidad estandarizada, que se puede utilizar para el desarrollo de software. Esta unidad, o contenedor, incluye la codificación y las dependencias de la aplicación para que pueda ejecutarse fácilmente en cualquier entorno informático.

Antes de Docker, las empresas solían utilizar máquinas virtuales (VM) para ejecutar aplicaciones. Éstas pueden emular ordenadores físicos, lo que permite a los de-

[13] Entorno de pruebas aislado que permite ejecutar programas o archivos sin afectar al sistema o plataforma en la que se ejecuta.

[14] Unidad estándar de software que empaqueta el código y todas sus dependencias para que la aplicación se ejecute de forma rápida y fiable en cualquier entorno informático.

sarrolladores convertir un servidor en varios. Sin embargo, este enfoque puede tener algunas desventajas.

Cada máquina virtual contiene una copia completa del sistema operativo y de la aplicación, así como los binarios y bibliotecas necesarios. Estos archivos pueden ocupar decenas de *GB* en un ordenador. Además, la virtualización del hardware para un sistema operativo (SO) invitado puede requerir una sobrecarga considerable.

En lugar de virtualizar el hardware, los contenedores virtualizan el **sistema operativo**. En Docker, los contenedores son abstracciones en la capa de aplicaciones que pueden contener tanto código como dependencias. Se puede ver una comparativa en la Figura 86.

En la misma máquina, varios contenedores pueden ejecutarse como procesos aislados:

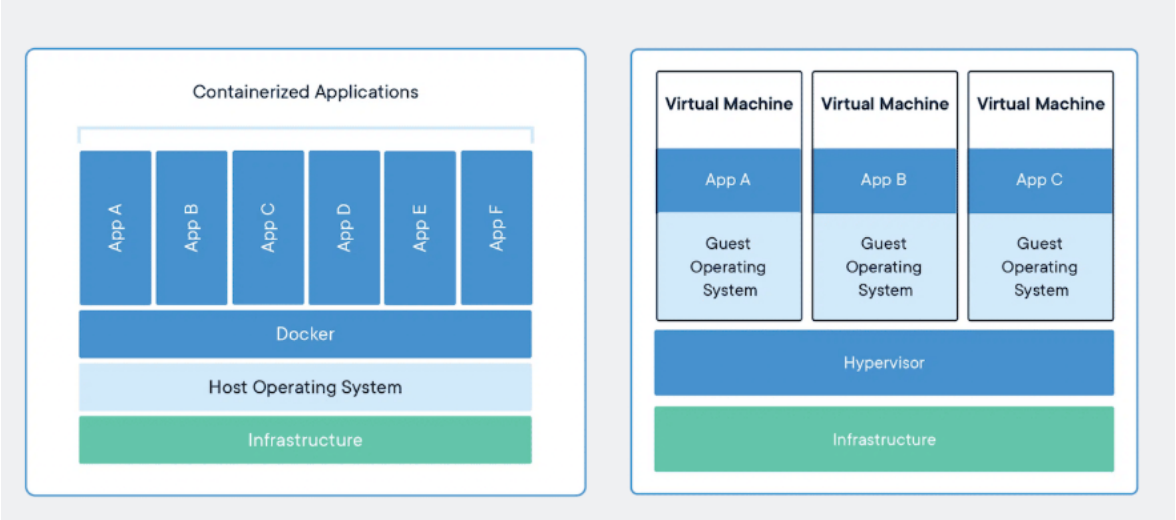


Figura 86: Comparativa entre Docker y una máquina virtual

Como resultado, los contenedores Docker suelen ocupar **menos espacio**. También pueden contener más aplicaciones sin necesidad de tantas máquinas virtuales y sistemas operativos.

Docker estandarizó el método de desarrollo y despliegue de software mediante contenedores. Utilizar contenedores permite el trabajo en distintos dispositivos o plataformas sin preocuparse por problemas de compatibilidad, ya que cada contenedor viene con las dependencias necesarias del sistema operativo donde se ejecutan.

Funciona mediante Docker Engine, la tecnología cliente-servidor para construir y contenerizar aplicaciones en Docker. Esencialmente, soporta todas las tareas relacionadas con la ejecución de tu aplicación basada en contenedores:

Los principales componentes del motor son:

- Docker Daemon: Gestiona las imágenes Docker, los contenedores, las redes y los volúmenes. También escucha las peticiones de la API de Docker y las procesa.

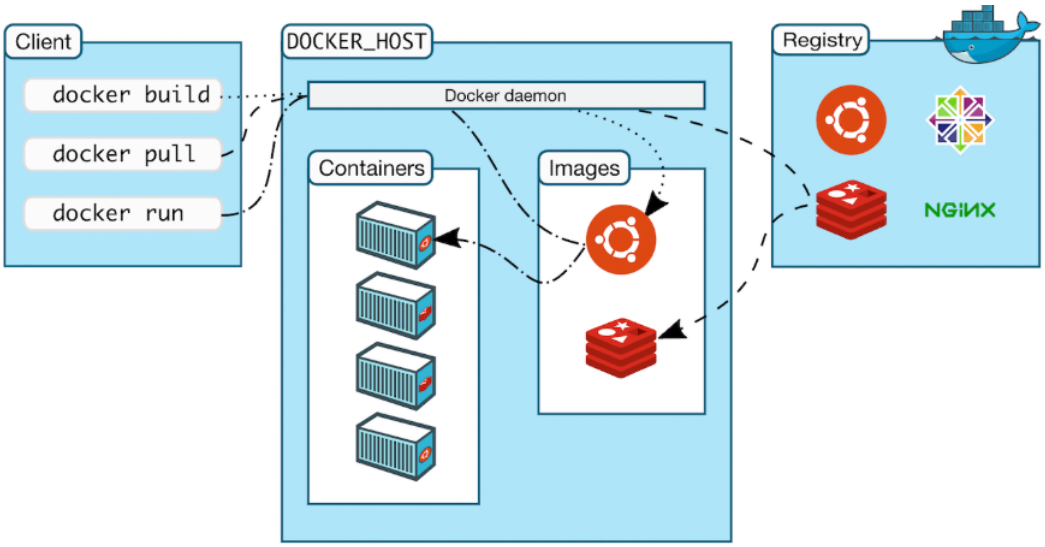


Figura 87: Docker Engine

- API REST del motor Docker.
- Docker CLI.

Dado que el proyecto requiere la ejecución simultánea de varios servicios (Mosquitto, Node-RED, MySQL, Home Assistant, Nginx y Grafana), gestionarlos uno por uno sería propenso a errores. Para solucionar esto, se utiliza la herramienta nativa **Docker Compose**.

Esta permite definir todo el "stack" de servicios en un único archivo de texto llamado `docker-compose.yml`). Este archivo actúa como una receta maestra que declara qué versiones usar, qué puertos abrir y, lo más importante, cómo se conectan los contenedores entre sí mediante una red interna virtual. En la Figura 88 se ve como se relacionan los contenedores entre sí y la estructura desde el hardware hasta el docker.

Las ventajas de este enfoque implementado son:

1. **Reproducibilidad:** Garantiza que el entorno sea idéntico tanto en las computadoras de desarrollo, las notebooks personales de los alumnos, como en el servidor de producción (la Raspberry Pi).
2. **Recuperación ante desastres:** En caso de falla crítica de la SD, se puede levantar toda la infraestructura compleja con un par de comandos.

8.3. Flujo de Trabajo y Automatización

Para profesionalizar el desarrollo y evitar la modificación de código "en caliente" sobre el equipo que controla la temperatura de la sala, se diseñó un ciclo de desarrollo dividido en dos entornos, aprovechando las capacidades de Git y Docker descritas anteriormente.

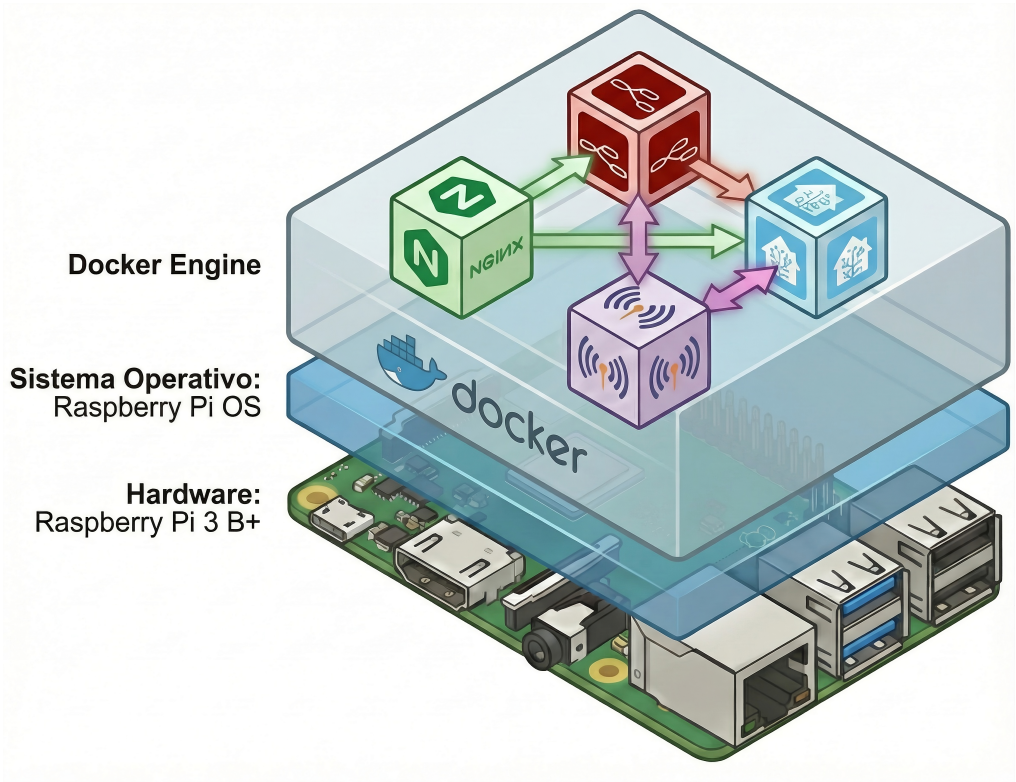


Figura 88: Estructura de contenedores interconectados definidos en el proyecto

8.3.1. Separación de Entornos

- **Entorno de desarrollo local:** Se ejecuta en la computadora personal. Aquí se programan los flujos, se prueban nuevas configuraciones de Home Assistant y se validan los cambios sin riesgo de afectar el sistema real.
- **Entorno de Producción (Remoto):** Es la Raspberry Pi instalada en la sala. Este entorno es de "solo lectura" respecto al código; solo recibe actualizaciones probadas.

8.3.2. Sincronización de Entornos

Para sincronizar ambos entornos de manera eficiente y segura, se desarrollaron *scripts* de automatización en consola (batch para Windows y bash para Linux). Estos reducen el proceso de actualización a la ejecución de un solo comando en cada extremo, eliminando la posibilidad de errores humanos al escribir comandos complejos repetitivamente.

1. En la notebook (`subir_cambios.bat`):

Este script gestiona la subida de nuevas versiones al repositorio. Al ejecutarlo:

1. Realiza chequeos de seguridad (verifica que Docker esté corriendo y que estemos en la carpeta correcta).

2. Detiene los contenedores locales (`docker compose down`) para asegurar que no haya archivos de base de datos en uso o bloqueados.
3. Verifica si hay cambios reales en el código mediante `git status`.
4. Si hay cambios, solicita un mensaje de *commit* al usuario (o genera uno automático con la fecha y hora).
5. Ejecuta la secuencia de Git (`add`, `commit`, `push`) para subir todo a la rama *main* del repositorio remoto.
6. Finalmente, vuelve a levantar el entorno local para seguir trabajando.

2. En la Raspberry Pi (`actualizar.sh`):

Este script se ejecuta en el servidor de producción para aplicar los cambios. Su lógica es "destructiva pero segura", priorizando que el sistema quede limpio y actualizado:

1. Detiene los contenedores en ejecución.
2. Asegura los permisos de los archivos (comando `chown`) para evitar conflictos con Git.
3. Fuerza la sincronización con el repositorio (`git fetch` y `git reset -hard`), descartando cualquier cambio local accidental para asegurar que el código sea idéntico al del repositorio.
4. Reconstruye y levanta los contenedores con el flag `-force-recreate`, lo que obliga a Docker a recrear los contenedores desde cero con las nuevas imágenes y configuraciones, eliminando cualquier estado residual (`-remove-orphans`).

El código fuente completo de estos scripts se encuentra detallado en los Anexos 13.3: *Script de Subida: `subir_cambios.bat` - Windows* y 13.4: *Script de Actualización: `actualizar.sh` - Raspberry Pi OS*.

8.3.3. Configuración del Docker

La arquitectura del sistema se define mediante el archivo `docker-compose.yml`. Este fichero establece que servicios se deben ejecutar, como se comunican entre sí y como se almacenan los datos de forma permanente.

Para garantizar la persistencia de la información ante reinicios del sistema, se utiliza la directiva `volumes`, que vincula las carpetas del sistema operativo anfitrión con el sistema de archivos de los contenedores.

El código completo y detallado de esta configuración se encuentra disponible en el Anexo 13.5: *Infraestructura de Contenedores*.

8.4. Implementación de la Lógica y Comunicaciones

La integración de los servicios descritos anteriormente se realiza mediante una arquitectura de flujos de datos definida principalmente en Node-RED y Mosquitto mediante MQTT.

8.4.1. Definición de Tópicos MQTT

Para ordenar la comunicación, se diseñó una jerarquía de tópicos estandarizada para el edificio que sigue la siguiente lógica: *edificio/piso/sala/variable*. En el proyecto se utilizan por ejemplo:

- `universidad/piso3/LIO/temperatura`: Publicado por el ESP de sensado. Contiene el valor actual en grados centígrados.
- `universidad/piso3/LIO/status`: Publicado por el ESP de sensado. Informa el estado operativo del dispositivo, permite diagnosticar si el nodo está operando ("Online"), o si se desconectó inesperadamente ("Offline").
- `universidad/piso3/LIO/temperatura_deseada`: Publicado por Home Assistant. Contiene la temperatura deseada por el usuario.

8.4.2. Flujo de Información

La lógica implementada sigue los siguientes pasos:

1. El **nodo sensor** mide la temperatura y la publica en el tópico correspondiente del broker Mosquitto. También publica su estado.
2. **Node-RED** se encuentra suscrito a este tópico mediante un *mqtt input*, recibe el dato, lo procesa y lo inyecta a Home Assistant.
3. Simultáneamente, el **controlador del actuador** (suscrito a los tópicos de temperatura y temperatura deseada) recibe los nuevos valores. Calcula el error ($e(t) = T_{deseada} - T_{med}$) y ajusta el ángulo del servomotor del difusor.
4. El usuario visualiza los cambios en tiempo real a través de Home Assistant, que está suscrito a todos los tópicos de estado para actualizar la interfaz gráfica.

Para la configuración de la temperatura deseada, se utiliza en Home Assistant un deslizador que permite variarla entre 15 y 30 grados en saltos de a 0.5 grados. Cuando el usuario desliza el control en la pantalla, se publica automáticamente el nuevo valor de referencia, cerrando el lazo de comunicación con el actuador físico.

8.4.3. Monitoreo de Disponibilidad: Last Will & Testament (LWT)

Es fundamental detectar instantáneamente si un nodo (sensor o actuador) deja de funcionar por cortes de energía o pérdida de conectividad. Para ello, se implementa el mecanismo de seguridad de MQTT conocido como **Last Will and Testament** (Última Voluntad).

El funcionamiento implementado es el siguiente:

1. Al conectarse al broker Mosquitto, el microcontrolador no solo envía sus datos, sino que también registra un mensaje de "testamento" en el tópico que acaba en `.../status` con el contenido "Offline".
2. El broker retiene este mensaje en memoria y **no lo publica** mientras la conexión se mantenga activa.
3. Inmediatamente después de conectar, el ESP publica manualmente un mensaje que dice "Online" (con flag *Retain*) en el mismo tópico, indicando al sistema que está operativo.
4. Si el dispositivo se desconecta abruptamente (por ejemplo, si se desenchufa la fuente de alimentación), el broker detecta la pérdida del enlace y publica automáticamente el mensaje "Offline" que tenía guardado.

Gracias a esto, la interfaz en Home Assistant puede alertar visualmente al usuario declarando al sensor como "No Disponible" u "Offline" sin necesidad de esperar a un timeout de lectura, permitiendo una gestión de fallos proactiva.

8.5. Interfaz de supervisióne interacción con el usuario

La interacción con el usuario final se centraliza en Home Assistant. Se diseñó un *dashboard* minimalista y moderno, accesible desde cualquier navegador web o dispositivo móvil conectado a la red.

Para lograr una estética profesional y una mejor experiencia de usuario, no se utilizaron las tarjetas predeterminadas, sino que se implementaron componentes personalizados mediante la librería **Mushroom Cards**. Esto permite crear una interfaz limpia, con botones redondeados (*chips*) y colores de estado.

8.5.1. Elementos del tablero

La interfaz muestra datos provenientes de entidades virtuales que actúan como nexo con el hardware. En la Figura 89 se detallan las vistas implementadas:

1. **Tarjeta de estado (con *Chips*):** En la parte izquierda se observa una tarjeta con etiquetas dinámicas creadas con *Mushroom Chips*. Estas indican instantáneamente:
   - **Nombre de la sala:** Su nombre con un icono distintivo de la sala.
   - **Estado del Difusor:** Indicador de color *verde/rojo* que muestra si el difusor está activo. Acompañado del texto *Abierto/Cerrado* dependiendo del estado.
   - **Conectividad:** Indicador *verde/rojo* que muestra si el sensor ESP01S está "Online" u "Offline", basado en el sistema LWT de MQTT.

   En la parte derecha se observa una tarjeta con los siguientes detalles de arriba a abajo:

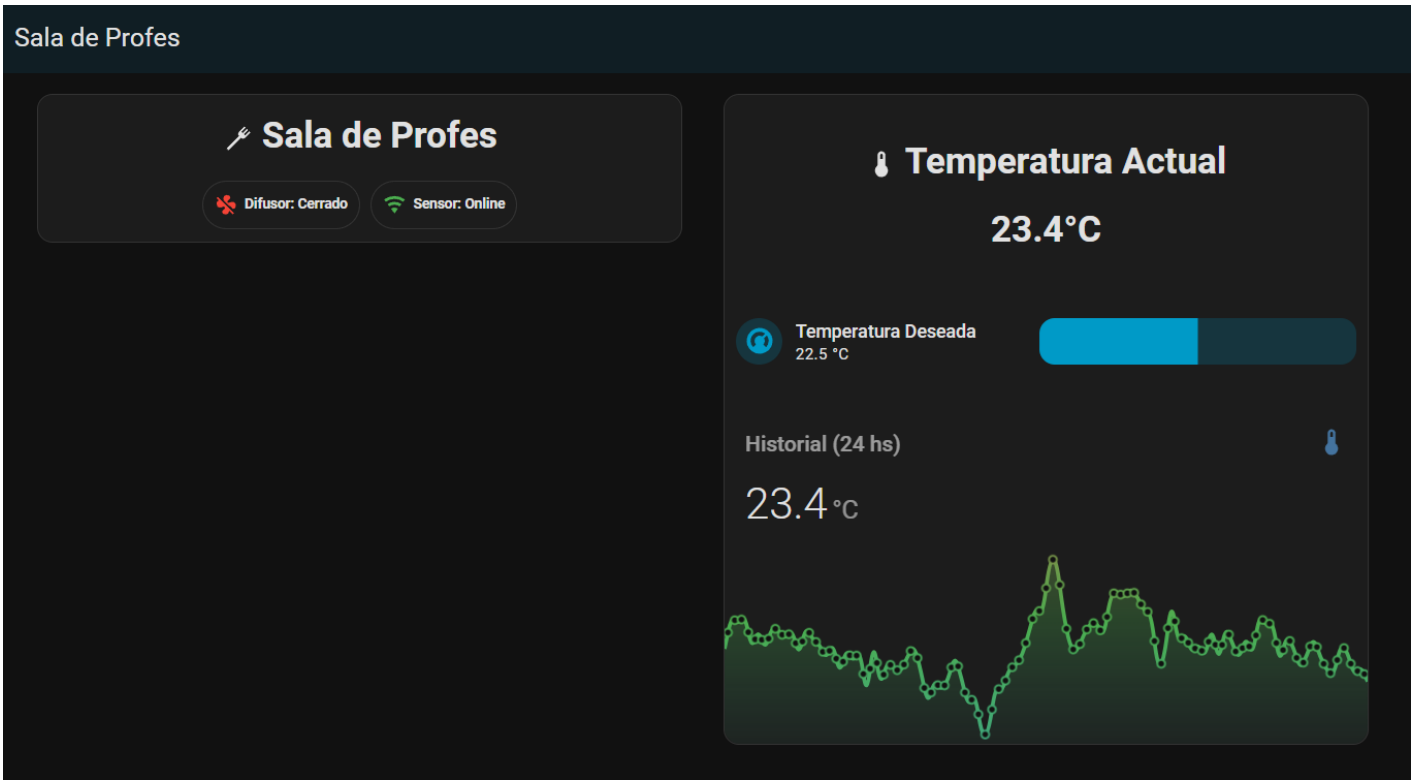


Figura 89: Vista del dashboard de una sala particular para el usuario final

2. **Indicador de temperatura actual:** Un texto de gran tamaño que visualiza el valor de la entidad `sensor.temperatura_lio`, actualizado en tiempo real vía Node-RED.
3. **Control de la temperatura deseada (*Slider*):** Vinculado al auxiliar `input_number.setpoint_lio`. Es un deslizador táctil que permite al usuario ajustar la temperatura deseada entre 15°C y 30°C en saltos de 0.5°C.
4. **Gráfico histórico:** Visible en la parte inferior, es una curva de tendencia generada con *Mini Graph Card* que permite analizar la evolución térmica de las últimas 24 horas, con interpolación de colores según la temperatura.

8.5.2. Lógica del *Backend*

La sincronización es gestionada por **Node-RED**. El flujo es de Home Assistant es bidireccional:

- **Sensor → Dashboard:** Node-RED recibe el dato crudo MQTT y actualiza el sensor virtual.
- **Dashboard → Actuador:** Al mover el slider, Node-RED detecta el cambio en el auxiliar y publica el nuevo valor en el tópico MQTT `.../temperatura_deseada`.

Figura 90: Vista general para el usuario final

8.5.3. Personalización y modo *Kiosco*

Para evitar manipulaciones indebidas en un entorno público, se instaló mediante **HACS**[15] el complemento **Modo Kiosco** (*Kiosk Mode*), el cual oculta las barras de menú y configuración, limitando al usuario exclusivamente al control de temperatura.

En la Figura 90 se puede dar un vistazo a la vista general que muestra a la izquierda la temperatura actual en el exterior del edificio y a la derecha una cuadrícula muestra las distintas habitaciones y sus temperaturas.

8.6. Accesibilidad y Resolución de Nombres

Para facilitar, tanto el acceso a los distintos servicios, como a la gestión remota sin necesidad de recordar direcciones IP numéricas (que pueden variar si el servidor DHCP del router asigna una nueva), se implementaron dos estrategias:

8.6.1. Resolución mDNS (Bonjour)

Se configuró el *hostname* de la Raspberry Pi como `fedami`. Gracias al protocolo mDNS (*Multicast DNS*), el sistema operativo difunde su nombre en la red local con el dominio `.local`.

Esto permite dos funcionalidades críticas:

1. **Acceso Web:** Cualquier dispositivo puede acceder al Dashboard ingresando:

```
http://fedami.local
```

[15] *Home Assistant Community Store*: Es un gestor de paquetes desarrollado por la comunidad que permite instalar integraciones y componentes de interfaz personalizado.s

2. **Gestión Remota (SSH):** Para tareas de mantenimiento, es posible conectarse a la terminal de la Raspberry Pi sin conocer su IP, utilizando el comando:

```
ssh pi@fedami.local
```

8.6.2. Enrutamiento por Proxy Inverso (Nginx)

Se utiliza **Nginx Proxy Manager** como puerta de enlace principal escuchando en el puerto 80. Su función es redirigir el tráfico web normal directamente a la interfaz de Home Assistant.

Para las herramientas de desarrollo e ingeniería (Node-RED, Mosquitto, Administración), se optó por mantener el acceso a través de sus puertos nativos pero aprovechando la resolución de nombre `fedami.local` para facilitar su ubicación.

El mapa de accesos definido es el presentado en la Tabla 1:

Rol	**Servicio**	**Dirección de Acceso**
Usuario Final	Home Assistant (Dashboard)	`http://fedami.local`
Técnico/Dev	Nginx Admin UI	`http://fedami.local:81`
	Node-RED (Flujos)	`http://fedami.local:1880`
	Mosquitto (Broker)	`tcp://fedami.local:1883`

Tabla 1: Mapa de accesos según el perfil de usuario.

De esta manera, se simplifica la experiencia para el usuario final (que solo debe recordar el nombre del equipo), mientras que los desarrolladores mantienen acceso directo a las herramientas de bajo nivel.

9. Simulaciones y mediciones

Para poder realizar las verificaciones de lo diseñado, lo que se realiza en la instancia previa a la implementación del sistema, una simulación en la que se pueda verificar que el sistema de control responda de forma correcta, para poder tener una validación del control calculado, y poder posteriormente implementarlo en la realidad. Cabe aclarar, que todas las simulaciones fueron realizadas en Simulink (Matlab), con la planta calculada y el controlador calculado.

En la etapa de mediciones, se instala el sistema en las bocas correspondientes, y se realizan mediciones durante todo un día, para verificar que el controlador funciones de manera correcta, y pueda interactuar con el usuario.

9.1. Simulación a lazo abierto

La simulación que se realiza a lazo abierto, implica una referencia de temperatura fija en un determinado valor. Se la lleva a cabo con una temperatura constante, para poder contrastar lo obtenido con el modelo teórico.

El bloque que permite la simulación de la planta es el Transfer Function, el bloque de ganancia de 1700 $\frac{m^3}{h}$ representa el caudal del Fancoil y los resultados se observan a través de un Scope:

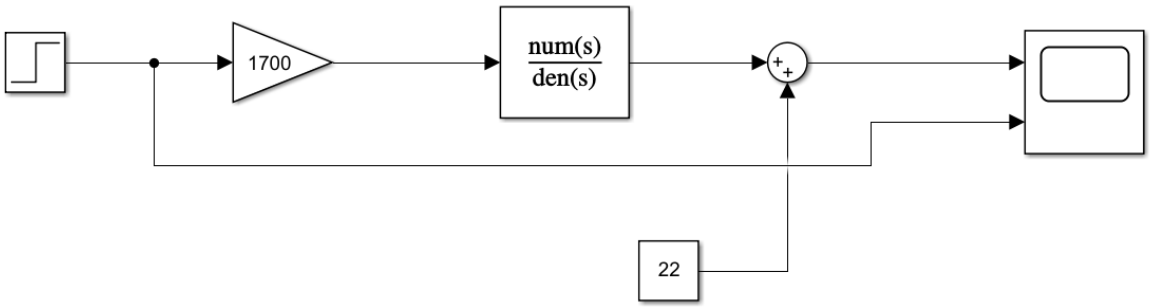


Figura 91: Simulación a lazo abierto - Simulink

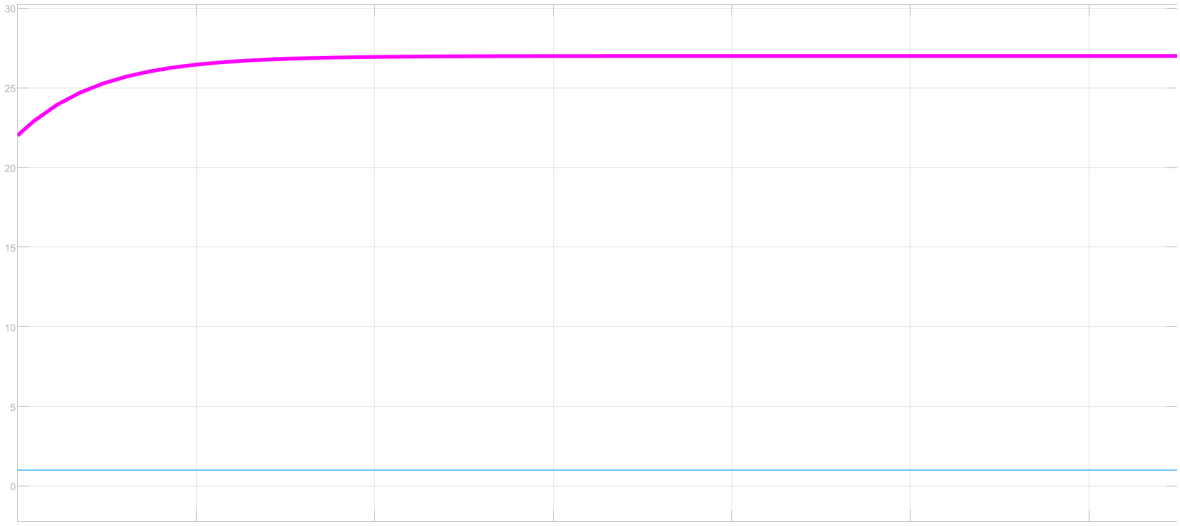


Figura 92: Resultado simulación a lazo abierto - Simulink

En el resultado de la simulación (Fig.92) se puede validar el comportamiento dinámico de la planta, ya que si se contrasta con el modelo calculado, se observa que concuerdan. El tiempo de simulación es propicio para poder observar la variación completa hasta que llega al valor final de estado estacionario.

9.2. Simulación a lazo cerrado

En este caso, ya se intercalan tanto el controlador como la saturación dada por el actuador. También, en esta simulación, se tiene en cuenta el retardo en la planta, para ver también su influencia en el lazo cerrado. El diagrama en bloques de esta simulación se presenta a continuación:

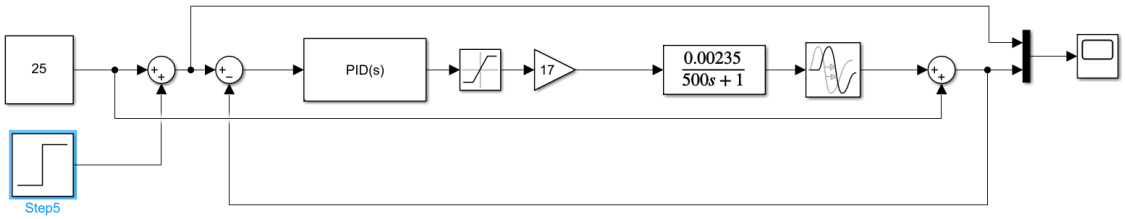


Figura 93: Simulación a lazo cerrado - Simulink

Aquí se observa la referencia en temperatura (que está dada por el escalón, los $25°C$ representa la temperatura ambiente), el bloque PID está seteado con los parámetros calculados (desviados un tanto para realizar una sintonización más fina), el bloque de saturación que simula la saturación en la región previa al 20 % de apertura del difusor y posterior al 70 %, la planta calculada y el bloque transport delay, para poder simular el retardo $e^{-}Ls$ que posee el sistema.

La realimentación se da desde la salida hacian el sumador/restador de la entrada que realiza la diferencia para proporcionarle al controlador la señal error.

Si se realizan simulaciones para distintas temperaturas de referencia (contemplando que la temperatura ambiente promedio en esta época del año ronda los $25°C$):

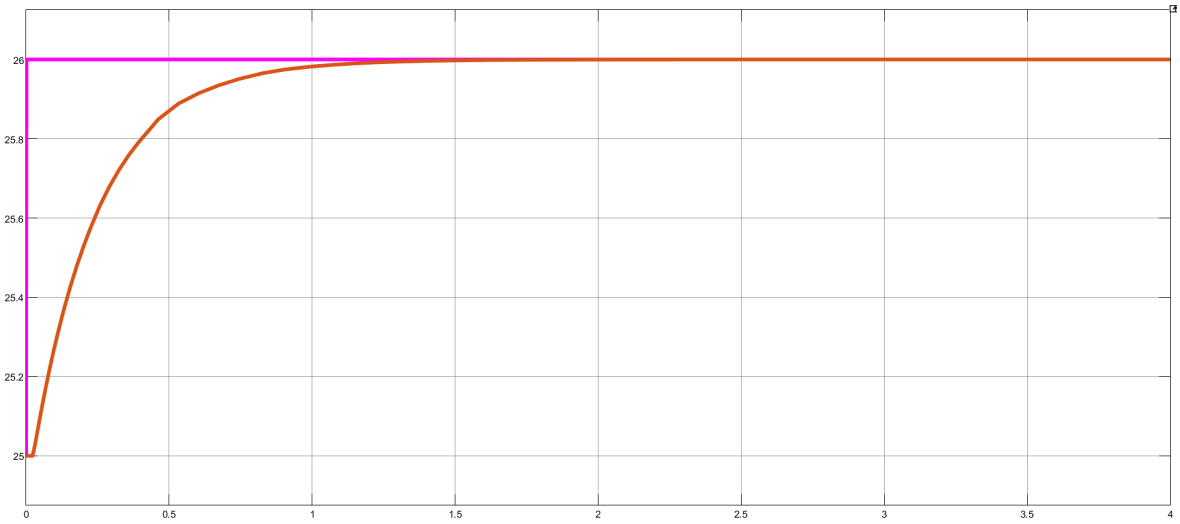


Figura 94: $T_{ref} = 28°C$

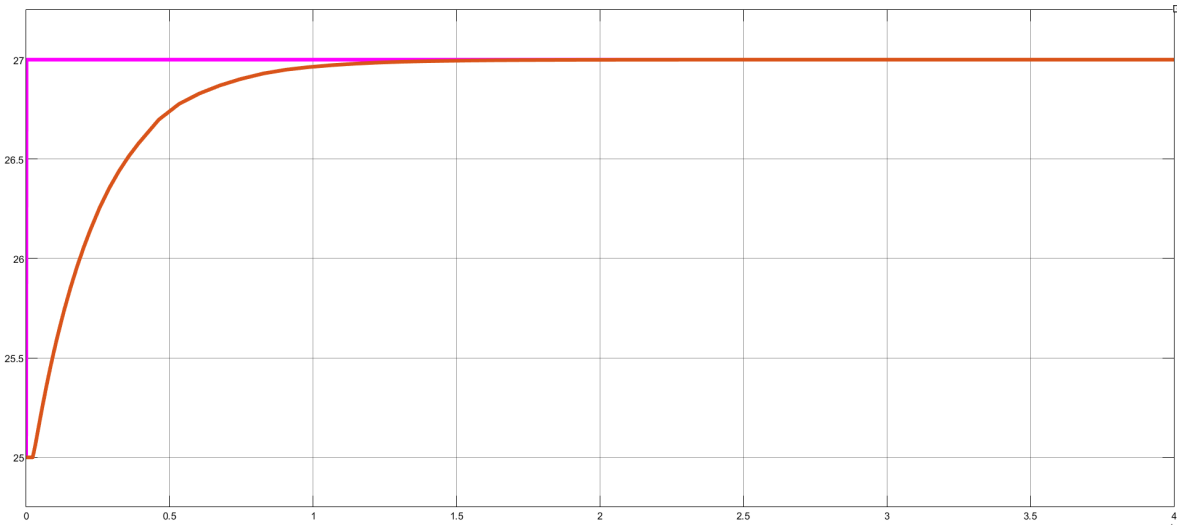


Figura 95: $T_{ref} = 28°C$

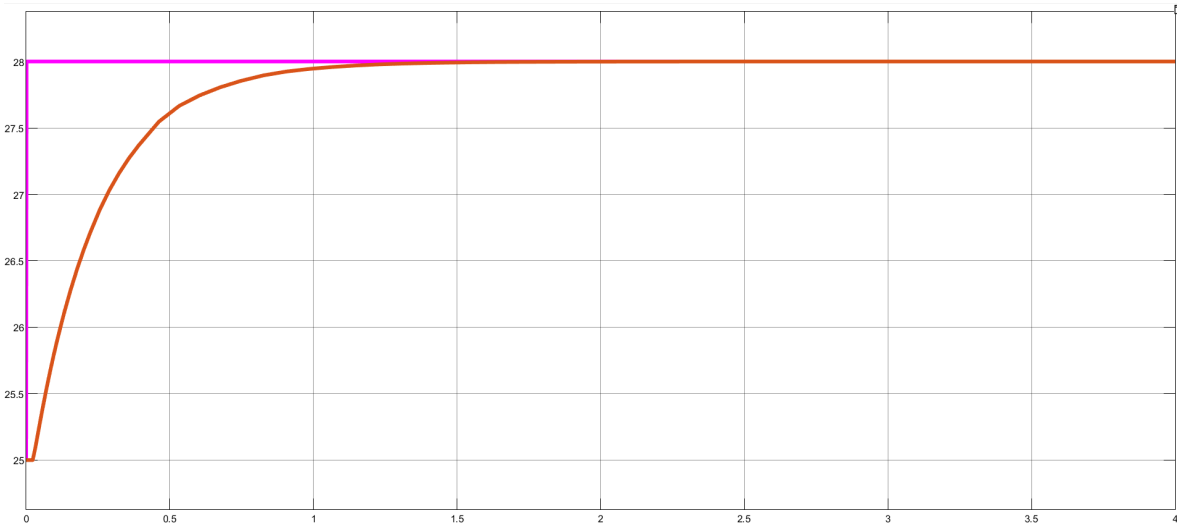


Figura 96: $T_{ref} = 28°C$

Se pueden observar en las Fig. **??**, **??** y **??**, que el controlador actúa de la forma esperada, y la salida logra igualar a la referencia con un bajo error de estado estacionario.

En la realidad, las perturbaciones en una habitación pueden ser relativamente frecuentes, por ejemplo al abrir una ventana de la habitación alteraría el comportamiento normal, o dejar la puerta abierta. Por lo tanto, se puede simular esta perturbación en el lazo de control, mediante por ejemplo:

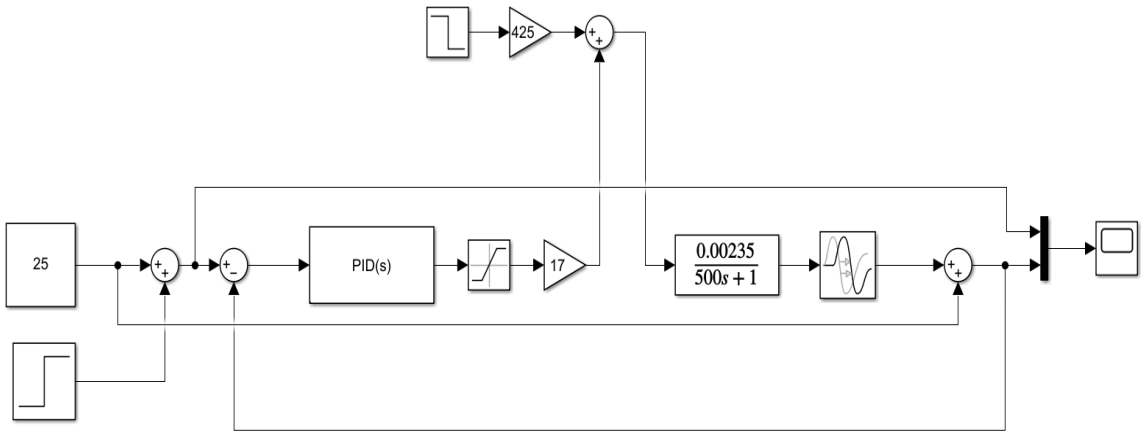


Figura 97: Lazo con una perturbación

Se configura una perturbación con un escalón negatico actuando en un momento en el cual la salida ya esté en su valor final , y se espera ver cómo se comporta el sistema ante la influencia de la misma:

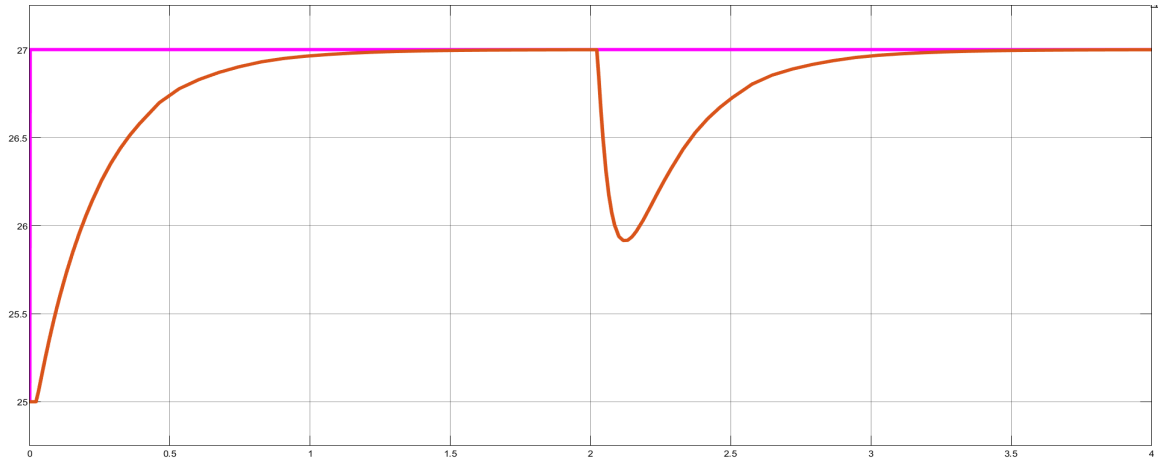


Figura 98: Rechazo de una perturbación - ejemplo: apertura de ventana que da a la sala de profesores

Se puede comprobar la efectividad del controlador a la hora de rechazar la perturbación, al poder restablecer el valor de la temperatura de salida al valor de referencia.

Cómo se desarrolla en el capítulo del control, se aplica una técnica de Anti-Windup para evitar que la acción integral siga creciendo cuando el actuador ya está saturado (Fig.99)

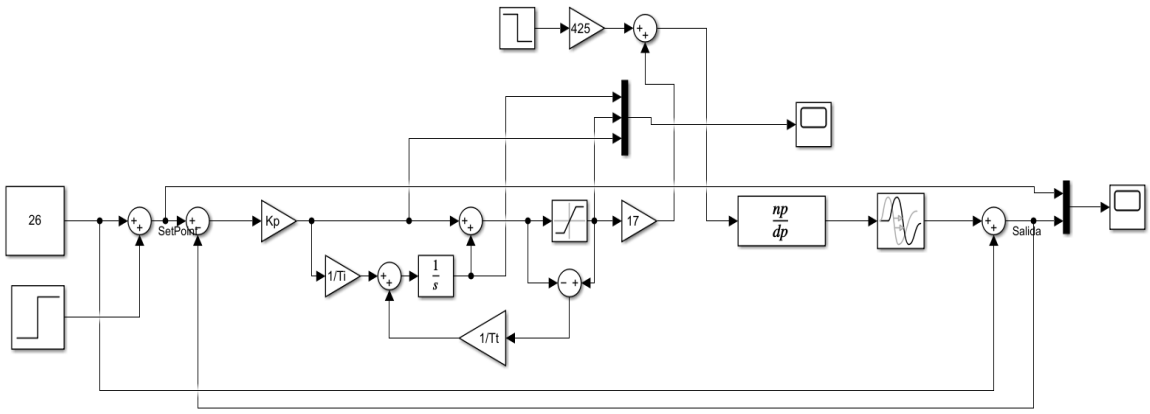


Figura 99: Aplicación anti-windup

Si se mide la salida del actuador, y la acción integral, se puede comprobar que el Anti-Windup cumpla con su cometido:

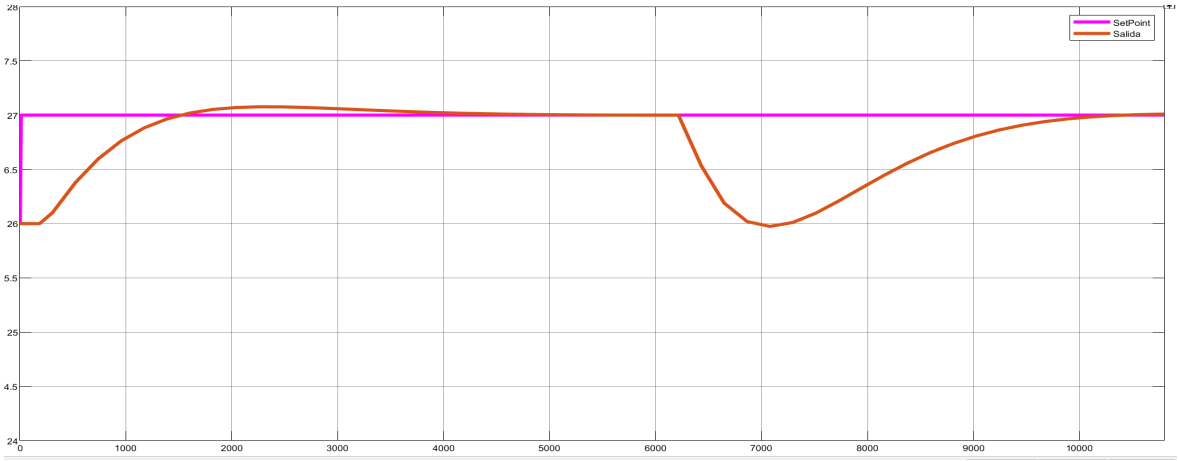


Figura 100: Scope - Salida

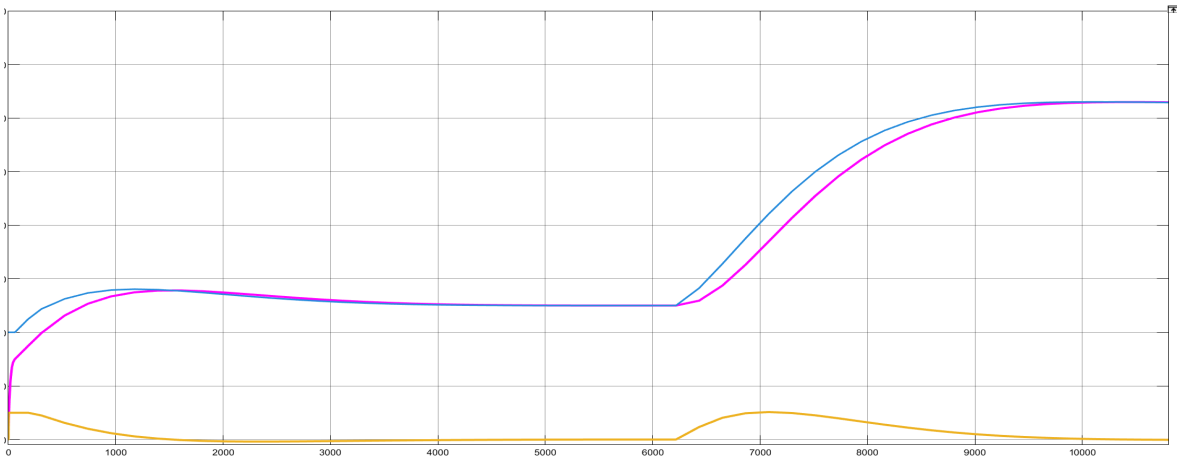


Figura 101: Scope - Señales internas

En la Fig.101 se tiene en azul la salida del actuador, en magenta la acción integral y en amarillo la acción proporcional. Se puede verificar que cuando el actuador satura, la acción integral deja de crecer y se establece en un valor, mientras que cuando no se encuentra saturado el sistema se comporta normalmente.

9.3. Hardware in the Loop

Para poder probar el controlador, sin el actuador todavía impreso, se realizó una técnica de simulación para probar sistemas embebidos al conectar el hardware real de control (el controlador corriendo en la ESP32) a una simulación virtual que representa los sensores y actuadores físicos. Esta técnica lleva el nombre de HIL (Hardwar-in-the-Loop - Hardware en el bucle), en el anexo se explica más en detalle. La simulación, en este caso tiene la siguiente estructura:

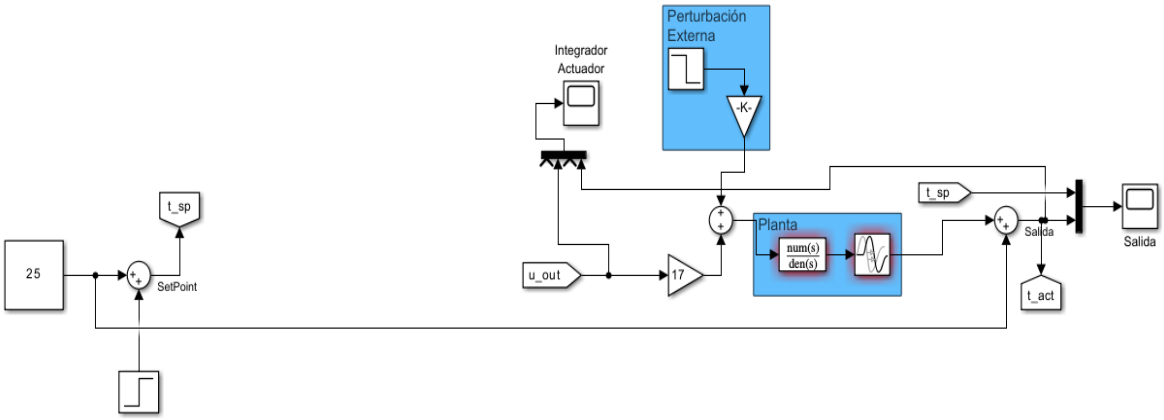


Figura 102: Simulación con emulador

se puede observar en la Fig.102 que no está intercalado el controlador, y se observan etiquetas de las señales:

- t_{act}: temperatura actual, medida a la salida de la planta.
- t_{sp}: es la temperatura del setpoint, que determina el usuario a su gusto.
- u_{out}: es la señal de control que proviene de la ESP32, en el que se encuentra el código del controlador, y se dirige al actuador físico.

Para poder realizar esta simulación, se utilizan los bloques de Simulink (se encuentran en la versión Matlab R2025):

- MQTT Client Suscribe.
- MQTT Client Publish.

los cuales se pueden observar debajo del lazo, integrados en un bloque llamado **Comunicación MQTT** (Fig.103):

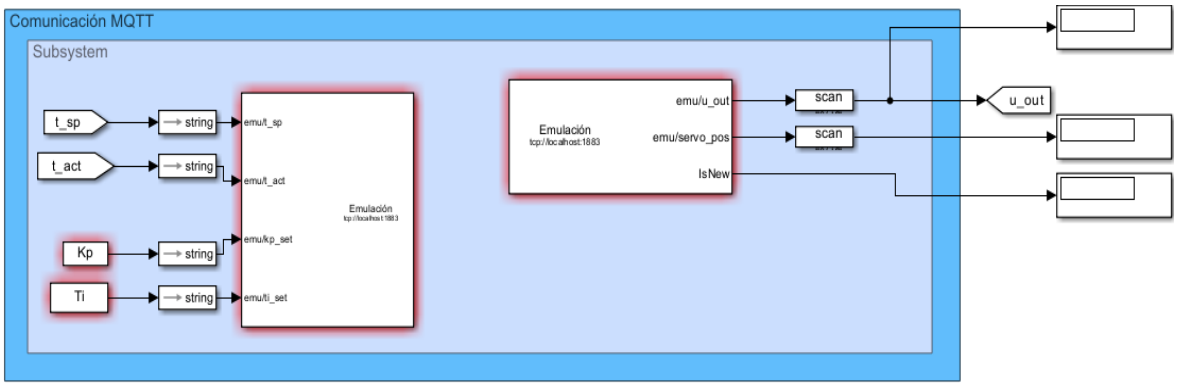
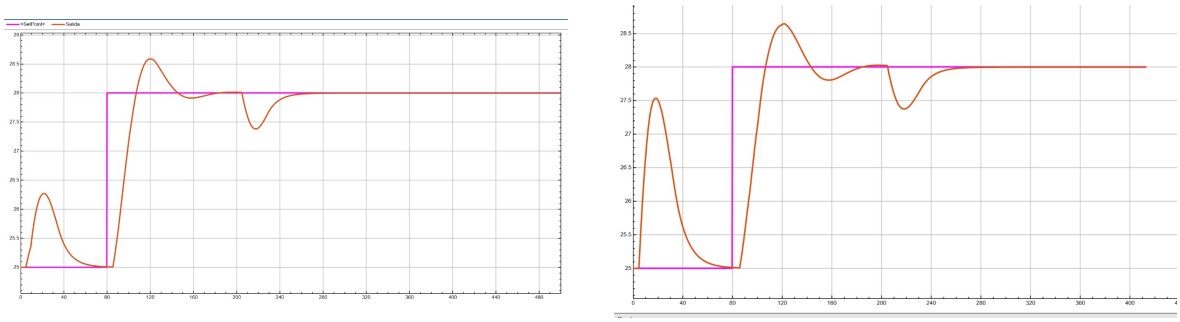


Figura 103: Comunicación MQTT

lo que hacen estos bloques es conectarse con el broker MQTT, donde se encuentran los datos del setpoint (seteado por el usuario) y la temperatura actual (subida por el

microcontrolador encargado de sensar). El otro artefacto que se conecta a este broker es la ESP32 que actúa de controlador, que es quien toma estos datos de temperaturas actual y de referencia, y realiza la cuenta para producir la señal u_{out}, que se dirige a la planta (la habitación).

Una vez configurada la simulación y cargado el código en el microcontrolador, se prueba el controlador, obteniéndose los siguientes resultados:



(a) Prueba 1

(b) Prueba 2

Figura 104: Respuestas simulación HIL

En estas mediciones, se puede ver algo importante y es que hay que reiniciar el controlador cada vez que se quiera realizar una medición ya que, como se observa en la Fig.104 se dan esos picos indeseados en la respuesta debido a que el microcontrolador queda con el estado anterior y no con el estado estacionario correspondiente.

Si se observa el dominio de tiempo, se puede ver que esta simulación se realiza en menos tiempo (en seg), pero la realidad es que para no estar tanto tiempo simulando, se realizó la simulación del sistema **acelerado** configurando tanto ladinámica de la planta como la del controlador de manera que sea más veloz, con un tiempo de establecimiento mucho menor.

También se realizó la simulación en tiempo real, lo cuál permitió ir monitoreando todas las señales presentes (105):

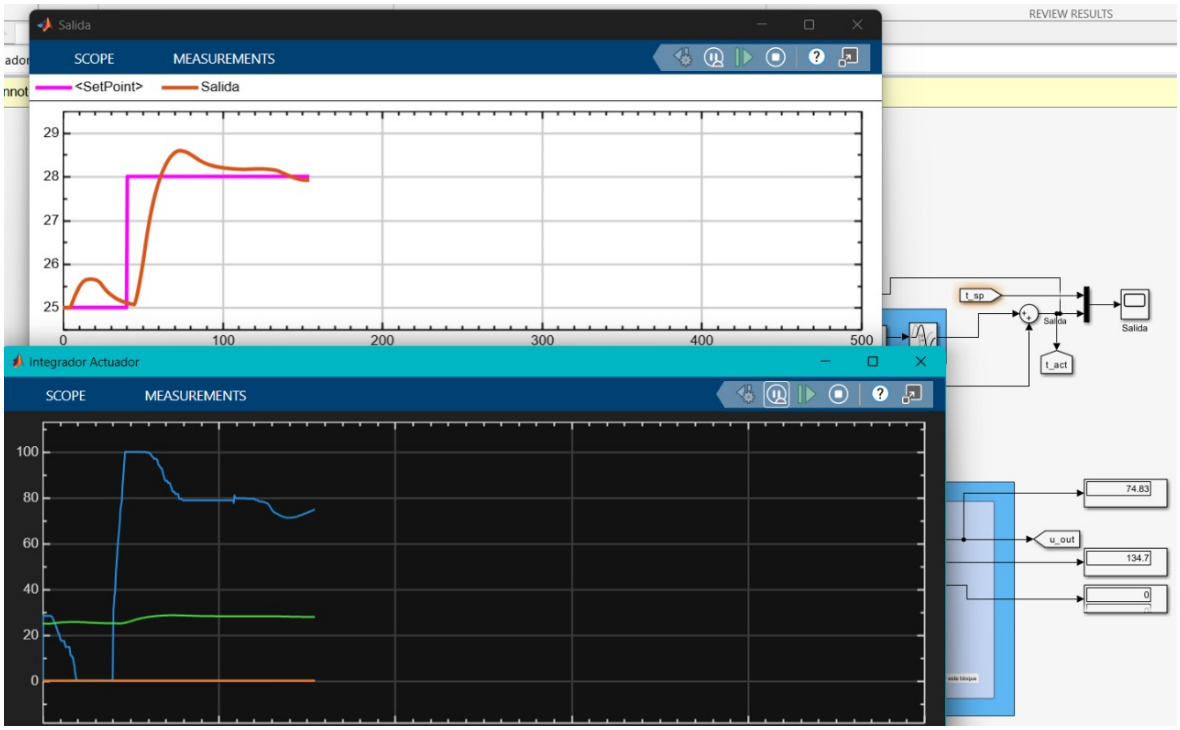


Figura 105: Simulación HIL en tiempo real

9.4. Mediciones

Una vez se tienen las simulaciones realizadas, el hardware electrónico programado y el difusor impreso, se proceden a realizar las pruebas en la habitación, y verificar que cumpla con el confort térmico dentro de la misma.

Se observa en las Figs., 106, 107,108 y 109, la implementación del sistema, amurando el difusor con la electrónica incorporada (controlador ESP32 + Reguladores de tensión + servomotor) en el techo. Mientras, a la altura de dónde las personas se encuentran sentadas, está la etapa sensora, con el sensor y la ESP32-C6-Zero programada de forma tal que se encuentre enviando la temperatura actual de la sala.

Los usuarios tienen la posibilidad de setear la referencia y poder monitorear el sistema desde el Home Assistant.

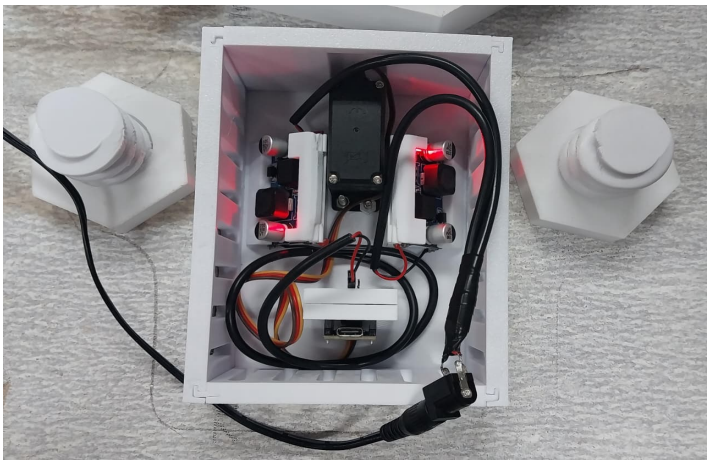


Figura 106: Electrónica interna

Figura 107: Caja vista desde afuera

Figura 108: Difusor amurado al techo - Cómo se ve en la realidad

Figura 109: Difusor actuando por orden del control

10. Trabajo Futuro

Este proyecto deja sentada las bases para posibles aplicaciones y desarrollos a futuro, como se detallan a continuación:

- *Implementación de módulo sensor*: Implementar el módulo sensor haciendo hincapié en la modularidad y eficiencia energética, haciendo que el módulo sea recargable y trabaje de forma intermitente utilizando el modo "sleep".
- *Implementación de red Zigbee*: Esto para una integración a un protocolo estandarizado en la industria, el cual podría mejorar los rendimientos energéticos de toda la infraestructura del sistema de comunicación.
- *Implementación modular de un piso*: El modelo es escalable para una aplicación a todo un piso.
- *Implementación de Gemelos Digitales*: Teniendo ya aplicado este sistema en un edificio se pueden implementar Gemelos Digitales para poder predecir comportamientos en la planta de interés.
- *Expansión del modelo matemático*: Se puede expandir el modelo matemático para poder obtener control ante distintos tipos de perturbaciones considerables dentro de un edificio, como lo pueden ser las temperaturas contiguas a la habitación, la humedad, la incidencia de luz.
- *Integración con Google Home*: Para una implementación mas integral sería optimo agregar a Goolge Home dentro del ecosistema, para que los usuarios tengan una mejor experiencia de usuario.

11. Conclusiones

En este trabajo, se diseñó un sistema de control de temperatura en el marco de las Smart Buildings, con conceptos de IoT. Tuvo etapas muy marcadas, como lo son las mediciones de la planta en estudio con el fin de obtener el modelo matemático que defina su comportamiento, el diseño del controlador con las técnicas de Control clásico vistas en la carrera, el diseño del actuador y la prueba de todo el sistema en la habitación de interés.

A lo largo de este proyecto final, se lograron utilizar muchas herramientas vistas en asignaturas de la carrera, desde Control, Comunicaciones, sistemas Digitales, hasta conceptos de Análisis Matemático para desarrollar la sección de la interpretación matemática del problema.

Además de conceptos vistos en la carrera, se tuvieron que aprender más herramientas, como lo son el protocolo MQTT y el uso del software para el diseño del actuador. Se realizó Ingeniería inversa para realizar las piezas, para las cuales hubo mucho estudio detrás, tanto para la estructura del diseño como para el uso del software Fusion 360 para su elaboración. Además, utilizamos impresión 3D para poder obtener un modelo físico real con el cual interactuar en las pruebas.

Se aprendieron también, técnicas novedosas de simulación, como lo es la simulación HIL, que no la habíamos visto en la carrera, y presentó una gran ventaja a la hora de probar el algoritmo de control que corría en la plataforma de hardware correspondiente.

Nos involucramos en todas las partes de un lazo de control:

- *Setpoint de referencia:* que seteaba un usuario a su gusto.
- *Temperatura actual:* la realimentación del lazo.
- *Actuador:* el actuador se diseñó y se probó, para poder ver su influencia en el lazo.
- *Modelo de la planta:* la planta era conocida, y nuestro objetivo era controlarla, para lo cual se hicieron ensayos en ella, con el fin de obtener un modelo matemático de la misma.

12. Anexo

12.1. Continuación del análisis matemático

12.1.1. Una expresión analítica para la función f(t)

Se supone que $f(t)$ se comporta de forma periódica con un período de un día, osea $24\,hs$ y que hay tres momentos del día que marcan una diferencia en el comportamiento de la temperatura. En la madrugada suele haber una temperatura más fresca que comienza a subir a lo largo de la mañana hasta que llega al valor máximo en el comiezo de la tarde (post-mediodía) y luego comienza a bajar hasta la mínima (que sería la madrugada siguiente). Un comportamiento de este tipo, se puede expresar como [13]:

$$f(t) = f_0 + f_1 \cos\left(\frac{\pi t}{12}\right) + f_2 \sin\left(\frac{\pi t}{12}\right) \tag{57}$$

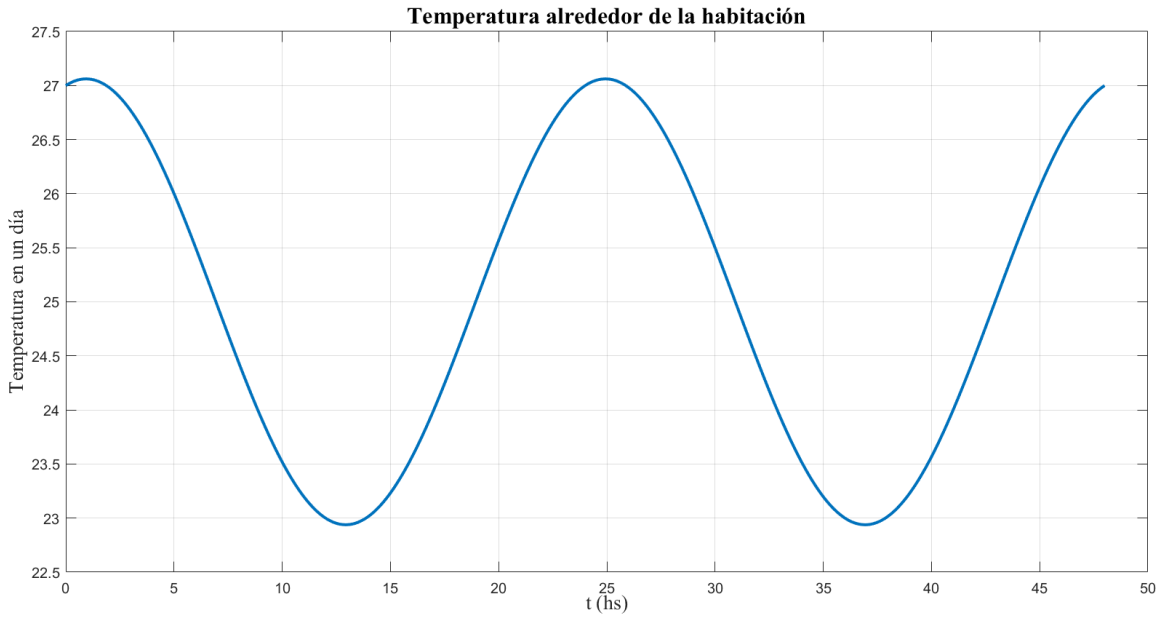


Figura 110: Variación de temperatura durante un par de días

si se conocen los valores máximo y mínimo de la temperatura en el día, y el valor de la misma en $t = 0$ se pueden determinar los valores de f_0, f_1 y f_2. Si se utiliza 65 en 43, se puede calcular la expresión analítica de los coeficientes:

$$\begin{aligned} C_n(t) = &\frac{\sqrt{2a}}{\pi n\left(1+\left(\frac{12\sigma\pi n^2}{a^2}\right)^2\right)}\left[\left(\frac{12\sigma\pi n^2}{a^2}f_2 + f_1\right)\left(e^{-\sigma\left(\frac{\pi}{a}\right)^2 n^2 t} - \cos\left(\frac{\pi t}{12}\right)\right)\right. \\ &\left. - \left(f_2 - \frac{12\sigma\pi n^2}{a^2}f_1\right)\sin\left(\frac{\pi t}{12}\right)\right] \end{aligned} \tag{58}$$

12.1.2. Cálculo de los coeficientes de la función

Si se toman tres temperaturas clave de distintas horas de un día cualquiera, ϕ_0=(temperatura al inicio del día, e $t = 0$), ϕ_{min}=(temperatura mínima en el día), ϕ_{max}=(temperatura máxima en el día) y sustituyendo en $f(t)$:

$$f(0) = f_0 + f_1\, cos(0) + f_2\, sen(0) = \phi_0$$
$$f_0 + f_1 = \phi_0 \tag{59}$$
$$f(6) = f_0 + f_1\, cos(\frac{6\pi}{12} + f_2\, sen(\frac{6\pi}{12}) = \phi_{min}$$
$$f_0 + f_1\, cos(\frac{\pi}{2}) + f_2\, sen(\frac{\pi}{2}) = \phi_{min}$$
$$f_0 + f_2 = \phi_{min} \tag{60}$$

$$f(14) = f_0 + f_1\, cos(\frac{14\pi}{12}) + f_2\, sen(\frac{14\pi}{12}) = \phi_{max}$$
$$f(14) = f_0 - \frac{f_1}{2} - \frac{f_1\sqrt{3}}{2} = \phi_{max} \tag{61}$$

Es un sistema de $3x3$, compatible determinado y que tiene por solución:

$$f_0 = \frac{2\,\phi_{max} + \phi_0\,\sqrt{3} + \phi_{min}}{3+\sqrt{3}}$$

$$f_1 = \phi_0 - \frac{2\,\phi_{max} + \phi_0\sqrt{3} + \phi_{min}}{3+\sqrt{3}}$$

$$f_2 = \phi_{min} - \frac{2\,\phi_{max} + \phi_0\,\sqrt{3} + \phi_{min}}{3+\sqrt{3}}$$

y así se obtiene:

$$f(t) = \frac{2\,\phi_{max} + \phi_0\,\sqrt{3} + \phi_{min}}{3+\sqrt{3}} + (\phi_0 - \frac{2\,\phi_{max} + \phi_0\sqrt{3} + \phi_{min}}{3+\sqrt{3}})\cos\left(\frac{t\pi}{12}\right) + (\phi_{min} - \frac{2\,\phi_{max} + \phi_0\,\sqrt{3} + \phi_{min}}{3+\sqrt{3}})\sin\left(\frac{t\pi}{12}\right) \tag{62}$$

12.1.3. Expresiones para la temperatura y el flujo de calor

Una buena aproximación para la distribución de temperatura a través de una pared de ancho a, hecha de material homogéneo con conductividad constante σ viene dada por la expresión [12]:

$$\overline{T}(x,t) = T(x,t) + H_0(x,t) \tag{63}$$

donde:

$$H_0(x,t) = (T_{int} - f(t))\frac{x}{a} + f(t) \tag{64}$$

con T_{int} la temperatura constante que se mantiene mediante una entrada/extracción de energía térmica en la habitación limitada por una pared y por:

$$f(t) = f_0 + f_1 \cos\left(\frac{\pi t}{12}\right) + f_2 \sin\left(\frac{\pi t}{12}\right) \tag{65}$$

que es la temperatura del lado de la cara exterior de la habitación (que puede ser calentada por el sol por ejemplo o una habitación vecina como es el caso).

Se puede utilizar la expresión para calcular $C_n(t)$ junto con la difusividad del material de las paredes de la habitación (Fig.111), para expresar:

$$T(x,t) = \sum_{n=1}^{N} C_n(t)\sqrt{\frac{2}{a}}\sin\left(\frac{n\pi x}{a}\right) + (T_{int} - f(t))\frac{x}{a} + f(t) \tag{66}$$

Material	Difusividad $(\frac{m^2}{s})$
Aluminio	97.5×10^{-6}
Hierro	22.8×10^{-6}
Mármol	1.2×10^{-6}
Hielo	1.2×10^{-6}
Concreto	0.75×10^{-6}
Ladrillo	0.52×10^{-6}
Vidrio	0.34×10^{-6}
Madera	0.13×10^{-6}
Corcho	0.038×10^{-6}
Lana de vidrio	0.023×10^{-6}
Lana de mineral de roca	0.022×10^{-6}
Poliestireno expandido	0.035×10^{-6}
Poliestireno Extruido	0.026×10^{-6}

Figura 111: Tabla de difusividad térmica para distintos materiales de construcción

Se puede encontrar una expresión para el flujo de calor en la pared interior ($x = a$). Como la solución depende solo de las variables x y t nos dice que, en caso de contacto perfecto, toda la pared se comporta igual que un bloque, ya que desaparecen condiciones de contorno (en y y en z). Para calcular el flujo de calor en $x = a$ a través de una pared de grosor a por metro cuadrado (medido en Wats o Vatios/m^2), al que se denota como $J_a(t)$, se tiene que [11]:

$$J_a(t) = \sigma \frac{\partial \overline{T}}{\partial x}(a, t) \tag{67}$$

Considerando las expresiones expuestas:

$$J_a(t) = \frac{\sigma\pi\sqrt{2}}{a\sqrt{a}} \sum_{n=1}^{\infty} (-1)^{n+1} n C_n(t) + \frac{T_{int} - f(t)}{a} \tag{68}$$

El flujo de calor a través de la cara interior de una pared de ancho $a > 0$, medido en $W/(m^2)$ se puede calcular en cualquier instante de tiempo t:

$$J_a(t) = -\frac{2\sigma}{a}\left(\frac{1}{1 + \left(\frac{12\sigma\pi}{a^2}\right)^2}\right)\left[\left(f_2 - \frac{12\sigma\pi}{a^2} f_1\right)\sin\left(\frac{\pi t}{12}\right) + \left(\frac{12\sigma\pi}{a^2} f_2 + f_1\right)\cos\left(\frac{\pi t}{12}\right)\right] + \frac{T_{int} - f(t)}{a} \tag{69}$$

Como se define $J_a(t) = \sigma \frac{\partial T}{\partial x}(a, t)$ y el vector unitario para una dirección positiva de x apunta hacia el interior de la habitación, entonces $J_a(t)$ denota el valor algebraico del flujo de calor que sale de la habitación, ósea si $J_a(t) > 0$ entonces el flujo de calor va de dentro hacia afuerza de la habitación y la habitación tiende a enfriarse, si $J_a < 0$ entonces el flujo de calor está dirigido en dirección contraria, osea de afuera de la habitacción hascia adentro.

Si $J_a(t) > 0$ la habitación tiende a enfriarse y si $J_a(t) < 0$ la habitación tiende a calentarse

Si se realizan análisis de signo de la expresión hallada para $J_a(t)$, se podrían determinas los momentos en el día dónde sería necesario añadir o extraer calor de una habitación, sólo es necesario poder definiri los coeficientes de la función analítica $f(t)$ para el entorno en dónde se encuentre la zona en estudio.

12.2. Fundamento de las relaciones entre engranajes

Se define paso de engranaje como el arco desde la cresta de un diente hasta la cresta del próximo. Sea a el paso de los engranajes, esta medida se mantiene para todo el juego de engranajes así se pueden acoplar de forma correcta unos con otros [15].

Si se calcula el perímetro del engranaje a partir del radio (R) a la cresta del diente:

$$P = 2\pi R \tag{70}$$

Entonces, la cantidad de dientes (Z) viene dada por la división del perímetro entre el paso del engranaje:

$$Z = \frac{2\pi R}{a} \tag{71}$$

La velocidad tangencial de un engranaje resulta de multiplicar la velocidad angular (o de giro) por el radio del enganaje.

$$v_T = \omega R \tag{72}$$

La potencia que transmite un engranaje motor es igual a la potencia que le entrega el eje donde está acoplado.

$$P = \omega M \tag{73}$$

Cuando se estudia la transmisión de un engranaje a otro, se considera que la potencia se transmite sin pérdidas. Existen pérdidas de potencia en la fricción de ejes y entre engranajes, pero se la considera despreciable.

$$P_M = P_C \tag{74}$$

En el punto de contacto entre engranajes la velocidad tangencial de uno y otro es la misma.

$$v_{TM} = v_{TC} \tag{75}$$

Calculando la relación entre la velocidad angular (o de giro) del engranaje motor y del engranaje conducido.

$$v_{TM} = \omega_M R_M \tag{76}$$

$$Z_M = \frac{2\pi R_M}{a} \rightarrow R_M = \frac{Z_M a}{2\pi} \tag{77}$$

$$v_{TM} = \frac{\omega_M Z_M a}{2\pi} \tag{78}$$

Como: $v_{TM} = v_{TC} \rightarrow \frac{\omega_M Z_M a}{2\pi} = \frac{\omega_C Z_C a}{2\pi}$, se obtiene:

$$\omega_M Z_M = \omega_C Z_C \tag{79}$$

Relación entre el momento del engranaje motor y el momento del engranaje conducido.

$$P_M = \omega_M M_M \tag{80}$$

$v_{TM} = \frac{\omega_M Z_M a}{2\pi} \rightarrow \omega_M = \frac{v_{TM} 2\pi}{Z_M a}$

$$P_M = \frac{v_{TM} 2\pi M_M}{Z_M a} \tag{81}$$

Como:

$$P_M = P_C \rightarrow \frac{v_{TM} 2\pi M_M}{Z_M a} = \frac{v_{TC} 2\pi M_C}{Z_C a}$$

Y como: $v_{TM} = v_{TC}$

$$\frac{M_M}{Z_M} = \frac{M_C}{Z_C} \tag{82}$$

12.3. Hardware en el Bucle (HIL)

La simulación de hardware en el bucle es una técnica para desarrollar y probar sistemas embebidos [16] Implica conectar las interfaces reales de entrada y salida (E/S) del hardware del controlador en un entorno virtual que simula el sistema físico. Las pruebas de hardware en el bucle se utilizan para validar la integración hardware-software y forman parte de los procesos de certificación en varias industrias. La principal ventaja de la simulación HIL es que permite probar tempranamente los algoritmos de control en el hardware,mitigando riesgos y acelerando el desarrollo, al permitir evaluar escenarios y probar la conectividad de E/S antes de que todos los componentes físicos estén disponibles, confirmando así la robustez del sistema sin comprometer equipos costosos.

La simulación HIL funciona mediante la interconexión del hardware del control real con un sistema físico simulado, conocido como la **planta**. Las conexiones entre el controlador real y la planta simulada son E/S analógicas y digitales reales. Suelen incluir protocolos de comunicación como UDP, TCP, CAN y otros estándares específicos de la industria. Las interfaces de comunicación, con sus configuraciones, temporización y cableado realies, son un componente clave de las pruebas HIL. Estos aspectos no se pueden replicar con precisión en simulaciones de modelo en el bucle (MIL) o software en el bucle (SIL). En la siguiente imagen, se observa una configuración típica de prueba de hardware en el bucle, el controlador real con interfaces de E/S reales está conectado al sistema en tiempo real:

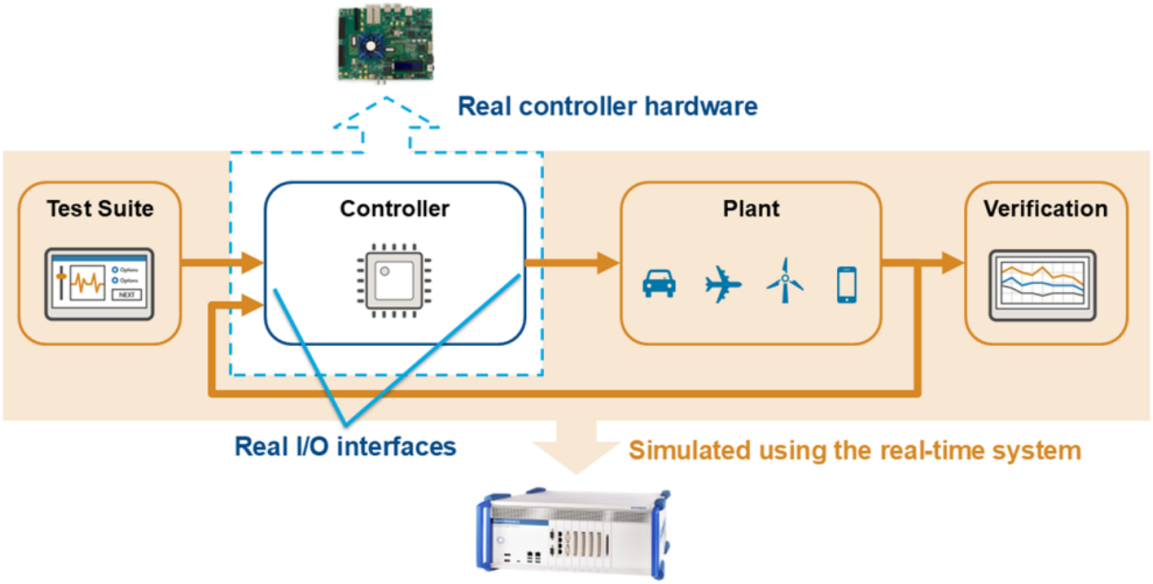


Figura 112: Hardware en el bucle

12.3.1. Pasos en la prueba HIL

- **Modelado de la planta:** el primer paso es crear un modelo matemático del sistema físico o la planta. Este modelo replica la dinámica del sistema real.
- **Simulación en tiempo real:** el modelo de la planta se ejecuta en un sistema de pruebas real, que simula el comportamiento de la planta en respuesta a las entradas del hardware del controlador. Esta simulación verifica que el sistema reaccione como lo haría en una situación real.
- **Conexión del controlador:** el hardware del controlador se conecta al simulador ent iempo real. Esta conexión permite al controlador interactuar con la planta virtual como si controlara el sistema real.
- **Adquisición de datos y retroalimentación:** el sistema adquiere continuamente datos del controlador y de la planta simulada. Estos datos se utilizan para retroalimentación del controlador, lo que le permite ajustar sus acciones en función de las respuestas simuladas de la planta. Los sistemas de prueba en tiempo real registran datos que pueden analizarse durante y después de la ejecución de la prueba de hardware en el bucle.
- **Pruebas y validación:** la simulación HIL permite realizar pruebas y refinamientos iterativos, lo que ayude a verificar que los algoritmos de control cumplan con su propósito.

12.3.2. Diferencias entre MIL, SIL, PIL y HIL

El modelo en el bucle (MIL), el software en el bucle (SIL), el procesador en el bucle (PIL) y el hardware en el bucle (HIL) son técnias de verificación y validación de

sistemas embebidos. Cada técnica umple una función específica en el ciclo de desarrollo, con diferentes niveles de fidelidad del entorno de simulación:

- **Modelo en el bucle (MIL):** el modelo en el bucle se usa en las etapas iniciales del desarrollo para verificar algortimos de control mediante un modelo de simulación de alto nivel. Este enfoque facilita el diseño y las pruebas iterativas de la lógica de control sin necesidad de hardware ni software. Tanto el controlador como las plantas físicas se simulan con los modelos correspondientes.
- **Software en el bucle (SIL):** el software en el bucle se centra en probar el código compilado del algoritmo de control en un entorno simulado. Este paso verifica que el software se comporte correctamente al ejecutarse, proporcionando un puente entre las simulaciones de modelos y las aplicaciones reales.

  SIL está evolucionando hacia pruebas virtuales, donde se simulan servicios adicionales del software del controlador. Este enfoque puede incluir la emulación de todo el sistema operativo del controlador embebido y la comunicación entre múltiples nodos que simulan componentes de la arquitectura orientada a servicios (SOA)
- **Procesador en el bucle (PIL):** el procesador en bucle implica la ejecución de los algoritmos de control en el procesador real o en un dispositivo similar conectado al entorno simulado en tiempo no real. Esta técnica detecta posibles problemas relacionados con la generación de código, el tiempo de ejecución y el comportamiento específico del procesador, confirmando así que el software funcionará correctamente en el hardware de destino.
- **Hardware en el bucle (HIL):** el hardware en el bucle (HIL) se utiliza para validar la integración de algoritmos de control con componentes de hardware reales. Al conectar el hardware del controlador real a una planta simulada en tiempo real, las pruebas HIL proporcionan un entorno de alta fidelidad para probar el sistema en condiciones y escenarios realistas.

12.4. Anemómetro UNIT

Un anemómetro mide la velocidad del viento usando distintos principios: los mecánicos (cazoletas/hélices) giran con el viento y cuentan revoluciones, los sónicos miden el tiempo de viaje de pulsos de sonido, y los de hilo caliente usan el enfriamiento de un cable caliente por el aire; el botón ünit.[en] modelos digitales sirve para cambiar la unidad de medida (m/s, km/h, nudos, etc.).

El anemómetro utilizado en el proyecto es mecánico (hélices) de la marca UNIT, el modelo es el **UT363**, que posee la siguiente ficha técnica:

ESPECIFICACIONES

	Rango	Resolución	Exactitud
Velocidad del viento	0～30m/s	0.1m/s	±5%rdg+0.5
Temperatura	-10～50°C	0.1°C	±2°C
	14～122°F	0.2°F	±4°F
Escala de viento	Nivel 0～12	1	±1
Tasa de muestreo			0.5s
Apagado automático			5min
Consumo eléctrico	En uso	mA	⩽25mA
	Apagado	uA	⩽10uA
Entorno laboral	Temperatura		0～40°C
	Humedad		⩽80%RH
Almacenamiento	Temperatura		-20～60°C
	Humedad		⩽75%RH
Alimentación	1.5V(AAA)x3		
Peso	118g		
Medidas	160x50x28mm		

Figura 113: Ficha técnica del anemómetro utilizado

13. Anexo II: Códigos

El código presentado ofrece una visión general de la implementación. Sin embargo, para acceder a los diagramas de flujo, las especificaciones de la interfaz, las explicaciones línea por línea, y cualquier información adicional ingrese al siguiente repositorio donde encontrará la documentación: Documentación

esto hay que cambiarlo al que se cargue en el repo

13.1. Actuador

```
#include <Arduino.h>
#include <ESP32Servo.h>
#include <WiFi.h>
#include <PubSubClient.h>
#include <math.h>

// ------------------------------------------------Definiciones de Hardware
static const int SERVO_PIN = 21;    // GPIO PWM apto para servo
static const int MIN_US    = 500;   // microsegundos min del servo
static const int MAX_US    = 2400;  // microsegundos max del servo
static const int SERVO_HZ  = 50;    // frecuencia típica de servo

Servo servo;

// ------------------------------------------------ LED integrado (un solo color)

// #ifndef RGB_BUILTIN
// static const int RGB_BUILTIN = RGB_BUILTIN;
// #endif

#define RGB_BRIGHTNESS 64
volatile uint32_t current_color = 0;

// LED ON si hay algún color distinto de negro
void setLedColor(uint8_t r, uint8_t g, uint8_t b) {
  current_color = ((uint32_t)r << 16) | ((uint32_t)g << 8) | (uint32_t)b;
  bool on = (r != 0) || (g != 0) || (b != 0);
  digitalWrite(RGB_BUILTIN, on ? HIGH : LOW);
}

// ------------------------------------------------ Controlador PI Discreto
// MISMOS parámetros que en el diseño / simulación de MATLAB
volatile float Kp = 5.0f;       // Kp de MATLAB
```

```
34 volatile float Ti = 250.0f;     // Ti de MATLAB [s]
35
36 // Tiempo de muestreo (Tm de MATLAB = 1/fs)
37 static const float Tm = 1.0f;  // 1 s (He vuelto a 1.0f para que
       la lógica de reset en muestra=5 sea 5 segundos)
38 const uint32_t CONTROL_INTERVAL_MS = (uint32_t)(Tm * 1000.0f);
39
40 // Rango en GRADOS del servo para el actuador físico
41 // Ajustá estos dos valores según tu mecanismo:
42 static const float SERVO_DEG_MIN = 0.0f;    // grados para mínima
       apertura
43 static const float SERVO_DEG_MAX = 180.0f;  // grados para má
     xima apertura
44
45 // Coeficientes del PI discreto (forma Tustin, igual a
     controladorZ)
46 volatile float Kpos;    // b0
47 volatile float Kneg;    // b1
48
49 // Variables de estado del controlador
50 volatile float u_logic        = 25.0f; // <--- VALOR DE
     POLARIZACIÓN (bias) INICIAL
51 volatile float pos_deg        = 00.0f; // posición actual del
     servo en grados
52 volatile float temp_setpoint = 25.0f; // SP
53 volatile float temp_actual   = 25.0f; // PV
54 volatile float error_k_1     = 0.0f;  // e[k-1]
55
56 // Contador de muestras para logging
57 volatile unsigned long sampleIndex = 0;
58
59 // Flag para el reseteo de estado inicial
60 bool initial_state_reset = false;
61
62 // ------------------------------------------------ WiFi / MQTT
63
64 static const char* WIFI_SSID = "Alumnos 2G";
65 static const char* WIFI_PASS = "";
66
67 static const char* MQTT_HOST = "FeDaMi.local";
68 static const uint16_t MQTT_PORT = 1883;
69 static const char* MQTT_USER = "";
70 static const char* MQTT_PASS = "";
71
72 // Tópicos de Suscripción para la Sintonización
73 static const char* MQTT_SUB_TOPIC_KP  = "emu/kp_set";
74 static const char* MQTT_SUB_TOPIC_TI  = "emu/ti_set";
75
76 // Tópicos de Suscripción para el Control
```

```
77  static const char* MQTT_SUB_TOPIC_SP  = "emu/t_sp";
78  static const char* MQTT_SUB_TOPIC_PV  = "emu/t_act";
79
80  // Tópicos de Publicación
81  static const char* MQTT_PUB_TOPIC_POS = "emu/servo_pos"; //
        Posición física (grados)
82  static const char* MQTT_PUB_TOPIC_U   = "emu/u_out";     //
        Mando lógico (0..100 %)
83
84  WiFiClient wifiClient;
85  PubSubClient mqtt(wifiClient);
86
87  // ================================================ FUNCIONES
        ================================================
88
89  // Escala el mando lógico (0..100 %) al rango de grados
        SERVO_DEG_MIN..SERVO_DEG_MAX
90  float scaleToDegrees(float u) {
91    if (u < 0.0f)   u = 0.0f;
92    if (u > 100.0f) u = 100.0f;
93
94    float span = (SERVO_DEG_MAX - SERVO_DEG_MIN);
95    return SERVO_DEG_MIN + (u / 100.0f) * span;
96  }
97
98  // Recalcula coeficientes del PI discreto (sin anti-windup),
99  // equivalentes a controladorZ = c2d(controlador,Tm,'tustin')
100 void recalculateCoefficients() {
101   if (Ti <= 0.0f || Kp <= 0.0f) {
102     Serial.println("[PI] ERROR: Kp o Ti no válidos. Coeficientes
          no actualizados.");
103     return;
104   }
105
106   const float T_factor = Tm / (2.0f * Ti);  // = 1/(2*Ti) con Tm
        =1
107
108   // coeficientes numéricos del PI en Tustin:
109   // C(z) = (b0 + b1 z^-1) / (1 - z^-1)
110   Kpos = Kp * (1.0f + T_factor);       // b0
111   Kneg = -Kp * (1.0f - T_factor);      // b1
112
113   Serial.printf("[PI] Kp=%.3f, Ti=%.3f, Tm=%.3f -> Kpos=%.6f,
        Kneg=%.6f\r\n",
114                 Kp, Ti, Tm, Kpos, Kneg);
115 }
116
117 void setServoDegrees(float deg) {
118   if (deg < SERVO_DEG_MIN) deg = SERVO_DEG_MIN;
```

```
119   if (deg > SERVO_DEG_MAX) deg = SERVO_DEG_MAX;
120
121   pos_deg = deg;
122   servo.write(int(180)-(int)roundf(deg));
123 }
124
125 static void publishState() {
126   if (mqtt.connected()) {
127     // Publicamos la posición del servo EN GRADOS
128     String s_deg = String(pos_deg, 1);
129     mqtt.publish(MQTT_PUB_TOPIC_POS, s_deg.c_str(), true);
130
131     // Publicamos el mando lógico U (0..100 %)
132     String s_u = String(u_logic, 3);
133     mqtt.publish(MQTT_PUB_TOPIC_U, s_u.c_str(), true);
134   }
135 }
136
137 // ---------------------------------PI discreto sin anti-windup,
      igual a lo que hace controladorZ en MATLAB
138 void applyControl() {
139   float e_k = temp_setpoint - temp_actual;
140
141   // u[k] = u[k-1] + Kpos*e[k] + Kneg*e[k-1]
142   float u_new = u_logic + (Kpos * e_k) + (Kneg * error_k_1);
143
144   // saturación 0..100 % lógico
145   if (u_new < 0.0f)   u_new = 0.0f;
146   if (u_new > 100.0f) u_new = 100.0f;
147
148   // actualizar estados
149   error_k_1 = e_k;
150   u_logic   = u_new;
151
152   // mapear 0..100 % a grados reales
153   float deg = scaleToDegrees(u_new);
154   setServoDegrees(deg);
155 }
156
157 // -------------------- Conexión MQTT --------------------
158
159 static bool connectMQTT() {
160   mqtt.setServer(MQTT_HOST, MQTT_PORT);
161   if (mqtt.connected()) return true;
162
163   String clientId = String("esp32c6-servo-") + String((uint32_t)
        ESP.getEfuseMac(), HEX);
164   Serial.printf("[MQTT] Conectando a %s:%u ...\r\n", MQTT_HOST,
        MQTT_PORT);
```

```
165
166   bool ok;
167   if (MQTT_USER && MQTT_USER[0] != '\0') {
168     ok = mqtt.connect(clientId.c_str(), MQTT_USER, MQTT_PASS);
169   } else {
170     ok = mqtt.connect(clientId.c_str());
171   }
172
173   if (!ok) {
174     Serial.printf("[MQTT] Fallo rc=%d\r\n", mqtt.state());
175     return false;
176   }
177
178   Serial.println("[MQTT] Conectado, suscribiendo tópicos...");
179
180   // Suscribiendo solo a los tópicos de entrada (SP, PV, Kp, Ti)
181   mqtt.subscribe(MQTT_SUB_TOPIC_SP, 1);
182   mqtt.subscribe(MQTT_SUB_TOPIC_PV, 1);
183   mqtt.subscribe(MQTT_SUB_TOPIC_KP, 1);
184   mqtt.subscribe(MQTT_SUB_TOPIC_TI, 1);
185   // El tópico U ("emu/u_out") ya no se suscribe, ahora es de
        publicación.
186
187   publishState();
188   return true;
189 }
190
191 // -------------------- MQTT Callback (texto o binario)
        ---------------------
192
193 static void mqttCallback(char* topic, byte* payload, unsigned
      int length) {
194   float value = NAN;
195
196   // 1) ¿Parece número en ASCII?
197   bool isAsciiNumber = true;
198   unsigned int n_check = length;
199   if (n_check > 64) n_check = 64;
200
201   for (unsigned int i = 0; i < n_check; ++i) {
202     char c = (char)payload[i];
203     if (!isDigit(c) && c != '.' && c != '-' && c != '+' &&
204         c != 'e' && c != 'E' && c != ' ') {
205       isAsciiNumber = false;
206       break;
207     }
208   }
209
210   if (isAsciiNumber) {
```

```
211     String s;
212     s.reserve(length);
213     for (unsigned int i = 0; i < length; ++i) s += (char)payload
            [i];
214     s.trim();
215     value = s.toFloat();
216     Serial.printf("[MQTT-CB] ASCII '%s' -> %f\r\n", s.c_str(),
            value);
217   }
218   else if (length >= 8) {
219     double d;
220     memcpy(&d, payload, 8);
221     value = (float)d;
222     Serial.printf("[MQTT-CB] Binario (>=8) -> double=%f (len=%u)
            \r\n", value, length);
223   }
224   else if (length >= 4) {
225     float f;
226     memcpy(&f, payload, 4);
227     value = f;
228     Serial.printf("[MQTT-CB] Binario (>=4) -> float=%f (len=%u)\
            r\n", value, length);
229   }
230   else {
231     Serial.printf("[MQTT-CB] Longitud muy corta (%u), no
            interpreto.\r\n", length);
232     return;
233   }
234
235   if (!isfinite(value)) {
236     Serial.println("[MQTT-CB] Valor no válido (NaN/Inf)");
237     return;
238   }
239
240   // --- Sintonización ---
241   if (strcmp(topic, MQTT_SUB_TOPIC_KP) == 0) {
242     if (value > 0.0f) {
243       Kp = value;
244       recalculateCoefficients();
245     }
246     return;
247   }
248   if (strcmp(topic, MQTT_SUB_TOPIC_TI) == 0) {
249     if (value > 0.0f) {
250       Ti = value;
251       recalculateCoefficients();
252     }
253     return;
254   }
```

```
255
256   // --- Control ---
257   if (strcmp(topic, MQTT_SUB_TOPIC_SP) == 0) {
258     temp_setpoint = value;
259     Serial.printf("[MQTT-CB] Nuevo SP = %.3f\r\n", temp_setpoint
          );
260   }
261   else if (strcmp(topic, MQTT_SUB_TOPIC_PV) == 0) {
262     temp_actual = value;
263     Serial.printf("[MQTT-CB] Nueva PV = %.3f\r\n", temp_actual);
264   }
265   else {
266     Serial.printf("[MQTT-CB] Tópico no manejado: %s\r\n", topic)
          ;
267   }
268 }
269
270 // ================================================ TAREAS
    FREERTOS ================================================
271
272 // ------------ Tarea de control discreto (PI + servo)
    ------------
273 void control_task(void * parameter) {
274   TickType_t xLastWakeTime;
275   const TickType_t xFrequency     = pdMS_TO_TICKS(
       CONTROL_INTERVAL_MS);
276   const TickType_t xLedPulseDelay = pdMS_TO_TICKS(5); // pulso
       LED corto
277
278   xLastWakeTime = xTaskGetTickCount();
279
280   while (true) {
281     vTaskDelayUntil(&xLastWakeTime, xFrequency);
282
283     if (WiFi.status() == WL_CONNECTED && mqtt.connected()) {
284
285       // --- LÓGICA DE RESET INICIAL (MANTENIDA) ---
286       // Se ejecuta una vez después de los primeros segundos
            para eliminar el transitorio de arranque.
287       // El sampleIndex se incrementa al final del bucle.
288       if (sampleIndex == 5 && !initial_state_reset) {
289         Serial.println("--- [PI Reset] Forzando u_logic = 50% y
              e[k-1] = 0.0 para inicio limpio ---");
290
291         // Resetea el integrador al valor de polarización (50%)
292         u_logic = 50.0f;
293
294         // Resetea el error anterior
295         error_k_1 = 0.0f;
```

```
296         initial_state_reset = true;
297
298         // Además, forzamos la PV a ser igual al SP para que el
                error sea cero justo antes del escalón
299         // Esto evita que el PI integre inmediatamente después
                del reset si la PV real es diferente de 25.0
300         temp_actual = temp_setpoint;
301       }
302       // ----------------------------------------
303
304       uint8_t r_prev = (current_color >> 16) & 0xFF;
305       uint8_t g_prev = (current_color >> 8)  & 0xFF;
306       uint8_t b_prev =  current_color        & 0xFF;
307
308       applyControl();
309       publishState();
310
311       // ----- LOG PARA MATLAB -----
312       unsigned long k = sampleIndex++;
313       float t = k * Tm;    // Tm = 1.0 s
314
315       Serial.printf("%lu,%.3f,%.3f,%.3f,%.3f,%.3f\r\n",
316                     k,
317                     t,
318                     temp_setpoint,
319                     temp_actual,
320                     u_logic,    // mando lógico 0..100 %
321                     pos_deg);   // posición en grados
322
323       // pulso de LED
324       setLedColor(0, 0, RGB_BRIGHTNESS);
325       vTaskDelay(xLedPulseDelay);
326       setLedColor(r_prev, g_prev, b_prev);
327     }
328   }
329 }
330
331 // ------------ Tarea de comunicaciones (WiFi + MQTT)
      ------------
332 void mqtt_task(void * parameter) {
333   uint32_t lastPrint = 0;
334   uint32_t lastReconnect = 0;
335
336   while (true) {
337     wl_status_t st = WiFi.status();
338
339     if (st != WL_CONNECTED) {
340       setLedColor(RGB_BRIGHTNESS, 0, 0);
341
```

```
342       if (millis() - lastPrint > 2000) {
343         Serial.printf("[WiFi] No conectado. status=%d\r\n", st);
344         lastPrint = millis();
345       }
346
347       if (millis() - lastReconnect > 5000) {
348         Serial.println("[WiFi] Reintentando WiFi.reconnect()");
349         WiFi.reconnect();
350         lastReconnect = millis();
351       }
352     }
353     else {
354       setLedColor(0, RGB_BRIGHTNESS, 0);
355
356       if (!mqtt.connected()) {
357         connectMQTT();
358       }
359       mqtt.loop();
360     }
361
362     vTaskDelay(pdMS_TO_TICKS(10));
363   }
364 }
365
366 //==================================================== SETUP y
      LOOP ====================================================
367
368 void setup() {
369   Serial.begin(115200);
370   delay(300);
371
372   Serial.println("\r\n--- ESP32-C6 - PI + MQTT + Servo (RTOS,
        grados, log CSV) ---");
373
374   // Header CSV para MATLAB
375   Serial.println("k,t,SP,PV,U_logic,Deg");
376
377   //pinMode(RGB_BUILTIN, OUTPUT);
378   setLedColor(0, 0, 0);
379
380   recalculateCoefficients();
381
382   servo.setPeriodHertz(SERVO_HZ);
383   servo.attach(SERVO_PIN, MIN_US, MAX_US);
384
385   // posición inicial: u_logic = 50 % -> grados intermedios
386   float deg0 = scaleToDegrees(u_logic);
387   setServoDegrees(deg0);
388
```

```
389   WiFi.mode(WIFI_STA);
390   WiFi.begin(WIFI_SSID, WIFI_PASS);
391   Serial.printf("[WiFi] Conectando a '%s'...\r\n", WIFI_SSID);
392
393   uint32_t t0 = millis();
394   while (WiFi.status() != WL_CONNECTED && (millis() - t0) <
          10000) {
395     delay(500); // NOTA: Este delay en setup() puede causar un
            WDT si la conexión tarda.
396     Serial.print('.');
397   }
398   Serial.println();
399
400   if (WiFi.status() == WL_CONNECTED) {
401     Serial.printf("[WiFi] Conectado. IP: %s\r\n", WiFi.localIP()
            .toString().c_str());
402     setLedColor(0, RGB_BRIGHTNESS, 0);
403   } else {
404     Serial.println("[WiFi] NO se pudo conectar en el arranque (
            timeout).");
405     setLedColor(RGB_BRIGHTNESS, 0, 0);
406   }
407
408   mqtt.setServer(MQTT_HOST, MQTT_PORT);
409   mqtt.setCallback(mqttCallback);
410
411   Serial.println("Inicializando tareas FreeRTOS...");
412
413   xTaskCreate(
414     control_task,
415     "ControlPI",
416     4096,
417     NULL,
418     3,
419     NULL
420   );
421
422   xTaskCreate(
423     mqtt_task,
424     "CommsMQTT",
425     8192,
426     NULL,
427     2,
428     NULL
429   );
430 }
431
432 void loop() {
433   // vacío: todo lo manejan las tareas
```

```
}
```

Listing 1: Controlador Final

13.2. Sensor

```
//
    ------------------------------------------------------------------

// CÓDIGO FINAL: ESP8266 + LWT (Offline Automático) + MQTT Diná
    mico
// SALA: Sala de Profes
//
    ------------------------------------------------------------------


#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <DHT.h>

//
    ==================================================================

// === BLOQUE DE CONFIGURACIÓN ===
//
    ==================================================================


// --- 1. Configuración WiFi ---
const char* ssid = "Alumnos 2G";
const char* password = "";

// --- 2. Configuración MQTT ---
const char* mqtt_host = "FeDaMi.local";
const int   mqtt_port = 1883;
const char* mqtt_client_id = "ESP01S_Sensor_Temp_Sala-de-Profes"
    ; // ID único

// --- 3. Configuración de Tópicos ---
// IMPORTANTE: Respetar Mayúsculas/Minúsculas igual que en Node-
    RED
String TOPICO_BASE = "universidad/piso3/sala_de_profes";

String TOPIC_TEMPERATURA = TOPICO_BASE + "/temperatura";
String TOPIC_HUMEDAD     = TOPICO_BASE + "/humedad";
String TOPIC_STATUS      = TOPICO_BASE + "/status";

// --- 4. Configuración del Sensor DHT ---
const int DHT_PIN = 2;     // GPIO2
```

```
#define DHT_TIPO DHT11

// --- 5. Tiempos ---
const long intervalo_publicacion = 60000; // 1 minuto

//
    ============================================================================


WiFiClient espClient;
PubSubClient client(espClient);
DHT dht(DHT_PIN, DHT_TIPO);
long ultimaPublicacion = 0;
char msgBuffer[10];
IPAddress mqtt_server_ip;

// --- Conexión WiFi ---
void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Conectando a WiFi: ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi conectado!");
  Serial.print("IP del ESP: ");
  Serial.println(WiFi.localIP());
}

// --- Buscar IP de la Raspberry ---
void resolver_ip_servidor() {
  Serial.print("Buscando IP de: ");
  Serial.println(mqtt_host);

  while (WiFi.hostByName(mqtt_host, mqtt_server_ip) == 0) {
    // CORRECCIÓN: Usamos la variable para que coincida con la
       realidad
    Serial.print("No se encuentra '");
    Serial.print(mqtt_host);
    Serial.println("'. Reintentando...");
    delay(2000);
  }

  Serial.print("¡Servidor encontrado! IP: ");
  Serial.println(mqtt_server_ip);
  client.setServer(mqtt_server_ip, mqtt_port);
```

```
}

// --- Reconexión MQTT con LWT (Last Will) ---
void reconnect_mqtt() {
  while (!client.connected()) {
    Serial.print("Intentando conexión MQTT...");

    // AQUI ESTA LA MAGIA DEL "OFFLINE" AUTOMATICO
    // Sintaxis: connect(ID, User, Pass, TopicoWill, QoS, Retain, MensajeWill)
    if (client.connect(mqtt_client_id, NULL, NULL, TOPIC_STATUS.c_str(), 1, true, "Offline")) {

      Serial.println("¡Conectado!");
      // Apenas nos conectamos, pisamos el mensaje "Offline" con un "Online" (con Retain=true)
      client.publish(TOPIC_STATUS.c_str(), "Online", true);

    } else {
      Serial.print("falló, rc=");
      Serial.print(client.state());
      Serial.println(" reintentando en 5s");
      delay(5000);
    }
  }
}

void setup() {
  Serial.begin(115200);
  dht.begin();
  setup_wifi();
  resolver_ip_servidor();
}

void loop() {
  if (!client.connected()) {
    reconnect_mqtt();
  }
  client.loop();

  long now = millis();
  if (now - ultimaPublicacion > intervalo_publicacion) {
    ultimaPublicacion = now;

    float h = dht.readHumidity();
    float t = dht.readTemperature();

    if (isnan(h) || isnan(t)) {
      Serial.println("Error sensor DHT11");
```

```
125         client.publish(TOPIC_STATUS.c_str(), "ErrorSensor");
126         return;
127       }
128
129       // Temperatura
130       snprintf(msgBuffer, 10, "%.1f", t);
131       client.publish(TOPIC_TEMPERATURA.c_str(), msgBuffer, true);
              // True para retain (opcional)
132       Serial.print("Temp: "); Serial.println(msgBuffer);
133
134       // Humedad
135       snprintf(msgBuffer, 10, "%.0f", h);
136       client.publish(TOPIC_HUMEDAD.c_str(), msgBuffer, true);
137       Serial.print("Hum: "); Serial.println(msgBuffer);
138     }
139 }
```

Listing 2: Codigo Sensor

13.3. Script de Subida: `subir_cambios.bat` - Windows

```
1  @echo off
2  setlocal EnableDelayedExpansion
3  TITLE Asistente Inteligente de Despliegue (Docker + Git)
4
5  :: --- CONFIGURACION ---
6  set RAMA=main
7
8  :: ============================================================
9  :: FASE 0: CHEQUEOS DE SEGURIDAD
10 :: ============================================================
11 :: Verificamos si Docker Desktop esta corriendo antes de
       intentar nada
12 docker info >nul 2>&1
13 if %errorlevel% neq 0 (
14     echo [ERROR] Docker no esta corriendo. Abortando.
15     pause
16     exit /b
17 )
18
19 :: ============================================================
20 :: FASE 1: PROTECCION DE INTEGRIDAD
21 :: ============================================================
22 echo.
23 echo [1/6] Deteniendo contenedores para asegurar integridad de
       BD...
24 docker compose down
25
26 :: ============================================================
```

```
:: FASE 2: DETECCION INTELIGENTE DE CAMBIOS
:: ============================================================
git status --porcelain > temp_git_status.txt
set size=0
for %%A in ("temp_git_status.txt") do set size=%%~zA
if exist temp_git_status.txt del temp_git_status.txt

if "%size%"=="0" (
    echo.
    echo    [!] El repositorio esta actualizado.
    goto :REINICIAR_DOCKER
)

:: ============================================================
:: FASE 3: GESTION DE VERSIONES (GIT)
:: ============================================================
echo.
echo [4/6] Preparando archivos...
git add .
set /p "mensaje=Mensaje del commit (Enter para auto): "
if "%mensaje%"=="" set "mensaje=Auto-update %date% %time%"

git commit -m "%mensaje%"
git push origin %RAMA%

:REINICIAR_DOCKER
:: ============================================================
:: FASE 4: RESTAURACION DEL ENTORNO
:: ============================================================
echo.
echo [6/6] Reactivando el Laboratorio...
docker compose up -d
```

Listing 3: Script de automatización para control de versiones y despliegue.

13.4. Script de Actualización: `actualizar.sh` - Raspberry Pi OS

```
#!/bin/bash

# Detener si hay errores
set -e

# Colores para feedback visual
VERDE='\033[0;32m'
AMARILLO='\033[1;33m'
SIN_COLOR='\033[0m'

# PASO 1: DETENER
```

```
echo -e "${AMARILLO}>>> 1. Deteniendo contenedores...${SIN_COLOR}"
docker compose down

# PASO 2: PERMISOS (Vital para que Git no falle)
echo -e "${AMARILLO}>>> 2. Asegurando propiedad de archivos...${SIN_COLOR}"
sudo chown -R $USER:$USER .

# PASO 3: GIT (Sincronizacion forzada)
echo -e "${AMARILLO}>>> 3. Sincronizando desde GitHub...${SIN_COLOR}"
git fetch --all
git reset --hard origin/main

# PASO 4: LEVANTAR CON FUERZA
echo -e "${AMARILLO}>>> 4. Reconstruyendo contenedores...${SIN_COLOR}"
# --force-recreate: Recrea el contenedor desde cero
# --remove-orphans: Elimina contenedores viejos
docker compose up -d --force-recreate --remove-orphans

echo -e "${VERDE}-----------------------------------------${SIN_COLOR}"
echo -e "${VERDE}   [ OK ] SISTEMA RECONSTRUIDO          ${SIN_COLOR}"
echo -e "${VERDE}-----------------------------------------${SIN_COLOR}"
```

Listing 4: Script para despliegue en producción.

13.5. Infraestructura de Contenedores

Este archivo define la versión de las imágenes utilizadas, la asignación de recursos de red y las políticas de persistencia de datos.

```
# Version de la sintaxis de Docker Compose
version: '3.8'

services:

  # ============================================================
  # SERVICIO 1: HOME ASSISTANT (Interfaz de Usuario y Control)
  # ============================================================
  homeassistant:
    # Imagen oficial estable desde el registro de contenedores de GitHub
    image: ghcr.io/home-assistant/home-assistant:stable
    container_name: homeassistant
```

```
13
14      # PERSISTENCIA DE DATOS:
15      # Vincula la carpeta local './homeassistant_config' con '/
            config'
16      # Esto asegura que los dashboards y configuraciones
            sobrevivan al reinicio.
17      volumes:
18        - ./homeassistant_config:/config
19
20      # GESTION DE RED:
21      # Expone el puerto 8123 para acceso web directo (http://
            localhost:8123)
22      ports:
23        - "8123:8123"
24
25      # VARIABLES DE ENTORNO:
26      # Define la zona horaria para correcta sincronizacion de
            historicos
27      environment:
28        - TZ=America/Argentina/Buenos_Aires
29
30      # POLITICA DE REINICIO:
31      # Arranca automaticamente al iniciar el sistema, salvo
            parada manual.
32      restart: unless-stopped
33
34  # ============================================================
35  # SERVICIO 2: NODE-RED (Motor de Logica y Flujos)
36  # ============================================================
37  nodered:
38      image: nodered/node-red:latest
39      container_name: node-red
40      # Ejecuta como root para tener permisos de escritura en
            volumenes montados
41      user: root
42
43      volumes:
44        - ./node-red-data:/data
45
46      ports:
47        - "1880:1880"
48
49      environment:
50        - TZ=America/Argentina/Buenos_Aires
51
52      restart: unless-stopped
53
54      # COMANDO DE INICIO PERSONALIZADO:
55      # 1. Instala dependencias NPM si faltan.
```

```
    # 2. Corrige permisos de archivos (chown) para evitar errores de acceso.
    # 3. Inicia el servicio Node-RED.
    entrypoint: ["/bin/sh", "-c"]
    command: >
      "echo '--  INICIANDO SECUENCIA DE ARRANQUE---  ' &&
       cd /data &&
       npm install --unsafe-perm --no-update-notifier --only=production &&
       echo '--  CORRIGIENDO PERMISOS---  ' &&
       chown -R 1000:1000 /data &&
       echo '--  EJECUTANDO NODE-RED---  ' &&
       cd /usr/src/node-red &&
       npm start -- --userDir /data"

  # ============================================================
  # SERVICIO 3: MOSQUITTO (Broker MQTT)
  # ============================================================
  mosquitto:
    image: eclipse-mosquitto:latest
    container_name: mosquitto

    # VOLUMENES DE CONFIGURACION:
    # - config: Archivo mosquitto.conf personalizado
    # - data: Persistencia de mensajes retenidos
    # - log: Archivos de registro para depuracion
    volumes:
      - ./mosquitto/config:/mosquitto/config
      - ./mosquitto/data:/mosquitto/data
      - ./mosquitto/log:/mosquitto/log

    ports:
      - "1883:1883" # Puerto MQTT estandar (TCP)

    restart: unless-stopped

  # ============================================================
  # SERVICIO 4: NGINX PROXY MANAGER (Gestion de Accesos Web)
  # ============================================================
  npm:
    image: 'jc21/nginx-proxy-manager:latest'
    container_name: nginx-proxy-manager
    restart: unless-stopped

    ports:
      - '80:80'    # Puerto HTTP estandar (Entrada principal)
      - '81:81'    # Interfaz de Administracion de Nginx
      - '443:443'  # Puerto HTTPS (SSL/TLS)
```

```
103     volumes:
104       - ./npm/data:/data
105       - ./npm/letsencrypt:/etc/letsencrypt
```

Listing 5: Configuración completa del stack Docker.

14. Anexo III: Hojas de datos

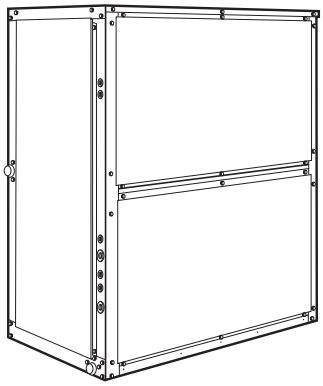
En este anexo se adjuntan las hojas de datos ("datasheets") de los componentes críticos seleccionados para el desarrollo del prototipo.

14.1. Fancoil del piso de electrónica - Marca Carrier

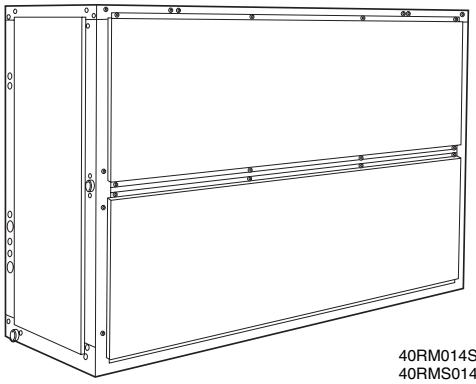
40RM/RMS
Packaged Air-Handling Units
50/60 Hz
6 to 30 Nominal Tons (21 to 105 kW)

Carrier
turn to the experts

Product Data



40RM007S-012S
40RMS008S-012S



40RM014S-034S
40RMS014S-034S

Features / Benefits

- Multiposition design can be installed horizontally or vertically without modification
- Standard double skin units for a better indoor air quality (IAQ)
- Polyester powder painted zinc coated steel panels provide additional protection against rusting & discoloration in areas with high UV factor
- High-static design meets a wider range of applications than competitive packaged air handler lines
- Economizer accessory provides ventilation air and ``free'' cooling
- Cooling coils with mechanically-bonded fins provide peak heat transfer
- Standard factory-installed thermostatic expansion valves (TXVs) on 40RM units
- Standard sloped drain pans
- 1" WASHABLE filters having frontal access to remove them
- Easy installation and maintenance; removal of one side panel allows access to serviceable components; external piping connections

The 40RM Series is your best choice for packaged air handlers. Model 40RM units have direct-expansion coils and model 40RMS units have chilled water coils. All models offer excellent fan performance, a unique combination of indoor air quality features, easy installation, and affordable prices. Their versatility and state-of-the-art features will provide economical performance now and in the future.

40RM-05PD 2011

Indoor air quality features:
The unique combination of features in the 40RM Series air handlers ensures that clean, fresh, conditioned air is delivered to the occupied space.

Cooling coils prevent the build-up of humidity in the room, even during part-load conditions. Unit sizes of 10 tons and above feature dual-circuit face-split coils.

1 inch (25.4 mm) WASHABLE air filters remove dust and airborne particles from the occupied space.

Standard double skin units provide an excellent resistance from the air-borne particles off the insulation thus maintaining an excellent indoor air quality (IAQ).

Pitched drain pan can be adjusted for a right- or left-hand connection to provide positive drainage and prevent standing condensate.

Accessory economizer can provide ventilation air to improve indoor air quality. When used with CO_2 sensors, the economizer admits fresh outdoor air to replace stale, recirculated indoor air.

Economy
The 40RM Series packaged air handlers have low initial costs, and they continue to save money by providing reduced installation expense and energy-efficient performance.

Quick installation is ensured by the multiposition design. Units can be installed in either the horizontal or vertical (upflow) configuration without modifications. All units have drain-pan connections on both sides, and pans can be pitched for right- or left-hand operation with a simple adjustment. Fan motors and contactors are prewired and TXVs are factory-installed on 40RM models.

High efficiency, precision-balanced fans minimize air turbulence, surging and unbalanced operation, thereby cutting operating expenses.

Economizer accessory precisely controls the blend of outdoor air and room air to achieve comfort levels. When the outside air enthalpy is suitable, outside air dampers can fully open to provide ``free'' cooling.

Rugged dependability
Polyester powder painted (zinc coated) panels provide excellent resistance to corrosion and also protect the colour in the areas where ultra-violet (UV) radiation factor is high.

Mechanically bonded coil fins provide improved heat transfer.

Galvanized steel fan housings are securely mounted to a die-formed galvanized steel deck.

Rugged pillow-block bearings (014-034 sizes) are securely fastened to the solid steel fan shaft with split collets and clamp locking devices. Smaller unit size shave spider-type bearings.

Coil flexibility
Model 40RM (direct-expansion coils) and 40RMS (chilled water coils) have galvanized steel casings; inlet and outlet connections are external & on the same end.

Chilled water coils have $^{1/2}$ in. (12.7 mm) diameter copper tubes mechanically bonded to aluminum sine-wave fins. All coils have non-ferrous headers.

Direct expansion (DX) coils are designed for use with Refrigerant 22 and have copper tubes mechanically bonded to aluminum sine-wave fins. Direct-expansion coils include matched, factory-installed thermostatic expansion valves (TXVs) with matching distributor nozzles.

Easier installation and service:
The multi-position design and component layout help you to get the unit installed and running quickly. The DX coils have factory-installed TXVs with matching distributor nozzles. Units can be converted from horizontal to vertical operation by simply repositioning the unit. Drain pan connections are duplicated on both sides of the unit. The filters, motor, drive, TXVs, and coil connections are easily accessed by removing a single side panels.

Table of contents

Model number nomenclature

4 0 R M - 0 1 6 S D B 5 0 1 E D

40RM Commercial Packaged Air Handler

Cooling Coil
- **-** Direct Expansion
- **S** Chilled Water

Nominal Capacity		Tons (kW)	
007	6 (21)	**016**	15 (52)
008	7-1/2 (26)	**024**	20 (70)
010	8-1/2 (29)	**028**	25 (87)
012	10 (35)	**034**	30 (105)
014	12-1/2 (43)		

S: SAMCO
D: Double Skin

Expansion Device
- **-** None (40RMS)
- **B** Thermostatic Expansion Valves (40RM)

Voltage Designation (V-Ph-Hz)
- **3** 208/230-1-60 (007-010 sizes only)
- **5** 208/230-3-60
- **6** 208/230/460-3-60 (all sizes 007-014; size 016 except YC and WD options)
  460-3-60 (sizes 016 with YC or WD options and all size 024-034 units)
- **8** 230-3-50
- **9** 400-3-50

Revision Number
- **0** Original Model

Factory-Installed Options

- **ED** Painted, Standard Motor and Standard Drive
- **FD** Painted, Standard Motor and Medium-Static Drive (Not available for 028 size for 60 Hz units and 016-028 for 50 Hz units)
- **RD** Painted, Alternate Motor and Medium-Static Drive (Available for 50 Hz sizes 016,024,028 and 60 Hz size 028 only)
- **WD** Painted, Alternate Motor and High-Static Drive

Packaging
- **1** Standard Domestic
- **3** Export

[2]YC and WD option codes for all 034 size units and 008, 010 units with 208/230-1-60 power designate standard motor and high-static drive.

NOTE: See the following table for the sizes available for each unit.

UNIT SIZE AVAILABILITY

UNIT	007	008	010	012	014	016	024	028	034
40RM	X	X		X	X	X	X	X	X
40RMS		X	X	X	X	X	X	X	X

QUALITY ASSURANCE
APPROVALS
ISO 9001:2008
EN ISO 9001:2008
ANSI/ASQC Q9001:2008
#0410019950420

40RM/RMS

Physical data

40RM - ENGLISH

UNIT 40RM	007SD	008SD	012SD	014SD	016SD	024SD	028SD	034SD
NOMINAL CAPACITY (Tons)	6	7½	10	12½	15	20	25	30
OPERATING WEIGHT (lb)								
Base Unit (Double skin) with TXV	448	452	472	804	819	824	1188	1198
Plenum	97	97	97	140	140	140	180	180
FANS								
Qty...Diam. (in.)	1...15	1...15	1...15	2...15	2...15	2...15	2...18	2...18
Nominal Airflow (cfm)	2400	3000	4000	5000	6000	8000	10,000	12,000
Airflow Range (cfm)	1800-3000	2250-3750	3000-5000	3750-6250	4500-7500	6000-10,000	7500-12,500	9000-15,000
Nominal Motor Hp (Standard Motor)*								
208/230-1-60	1.3	2.4	-	-	-	-	-	-
208/230-3-60 and 460-3-60	2.4	2.4	2.4	2.9	3.7	5.0	7.5	10.0
575-3-60	1.0	2.0	2.0	3.0	3.0	5.0	7.5	10.0
230-3-50, 400-3-50	2.4	2.4	2.9	2.9	2.9	5.0	7.5	10.0
Motor Speed (rpm)								
208/230-1-60	1725	1725	-	-	-	-	-	-
208/230-3-60 and 460-3-60	1725	1725	1725	1725	1725	1745	1745	1745
575-3-60	1725	1725	1725	1725	1725	1745	1755	1755
230-3-50, 400-3-50	1425							
REFRIGERANT	R-22							
Operating charge (lb) (approx per circuit)[2]	3.0	3.0	1.5/1.5	2.0/2.0	2.5/2.5	3.5/3.5	4.5/4.5	5.0/5.0
DIRECT-EXPANSION COIL	Enhanced Copper Tubes, Aluminum Sine-Wave Fins							
Max Working Pressure (psig)	435							
Face Area (sq ft)	6.67	8.33	10.0	13.25	17.67	19.88	24.86	29.83
No. of Splits	1	1	2	2	2	2	2	2
Split Type...Percentage	-	-	Face...50/50					
No. of Circuits per Split	12	15	9	9	12	13	15	18
Rows...Fins/in.	3...15	3...15	3...17	3...15	3...15	3...17	3...15	3...15
STEAM COIL**								
Max Working Pressure (psig at 400 F)	175							
Total Face Area (sq ft)	6.67	6.67	6.67	13.33	13.33	13.33	15.0	15.0
Rows...Fins/in.	1...9	1...9	1...9	1...10	1...10	1...10	1...10	1...10
HOT WATER COIL**								
Max Working Pressure (psig)	150							
Total Face Area (sq ft)	6.67	6.67	6.67	13.33	13.33	13.33	15.0	15.0
Rows...Fins/in.	2...8.5	2...8.5	2...8.5	2...8.5	2...8.5	2...8.5	2...12.5	2...12.5
Water Volume								
(gal)		8.3			13.9		14.3	
(ft^3)		1.1			1.85		1.90	
PIPING CONNECTIONS								
Quantity...Size (in.)								
DX Coil - Suction (ODF)	1...1 1/8	1...1 1/8	2...1 1/8	2...1 1/8	2...1 1/8	2...1 1/8	2...1 3/8	2...1 3/8
DX Coil - Liquid Refrigerant (ODF)	1...5/8					2...5/8		
Steam Coil, In (MPT)	1...2 1/2					1...2 1/2		
Steam Coil, Out (MPT)	1...1 1/2					1...1 1/2		
Hot Water Coil, In (MPT)	1...1 1/2		1...1 1/2			1...2		
Hot Water Coil, Out (MPT)	1...1 1/2		1...1 1/2			1...2		
Condensate (PVC)	1...1 1/4 ODM/1 IDF							
FILTERS	Washable - Factory Supplied							
Quantity...Size (in.)	4...16 x 24 x 1			4...16 x 20 x 1 4...16 x 24 x 1			4...20 x 24 x 1 4...20 x 25 x 1	
Access Location	Front							

LEGEND

DX - Direct Expansion
TXV - Thermostatic Expansion Valve

*Refer to alternate Fan Motor Data table, pages 52 and 53, for alternate motor data.
[2]Units are shipped without refrigerant charge.
**Please contact your local Carrier sales office for availability.
Please contact your local Carrier sales office for the weights of single skin units.

40RM - SI

UNIT 40RM	007SD	008SD	012SD	014SD	016SD	024SD	028SD	034SD
NOMINAL CAPACITY (kW)	21	26	35	43	52	70	87	105
OPERATING WEIGHT (kg)								
Base Unit (Double skin) with TXV	203	205	214	364	371	373	538	542
Plenum	44	44	44	63	63	63	82	82
FANS								
Qty...Diam. (mm)	1...381	1...381	1...381	2...381	2...381	2...381	2...457	2...457
Nominal Airflow (L/s)	1133	1604	1888	2360	2831	3775	4719	5663
Airflow Range (L/s)	850-1416	1203-2006	1416-2360	1770-2949	2124-3539	2831-4719	3539-5899	4247-7079
Nominal Motor kW (Standard Motor)*								
208/230-1-60	0.97	1.79	-	-	-	-	-	-
208/230-3-60 and 460-3-60	1.79	1.79	1.79	2.16	2.76	3.73	5.59	7.46
575-3-60	0.75	1.49	1.49	2.24	2.24	3.73	5.59	7.46
230-3-50, 400-3-50	1.79	1.79	2.16	2.16	2.16	3.73	5.59	7.46
Motor Speed (r/s)								
208/230-1-60	28.8	28.8	-	-	-	-	-	-
208/230-3-60 and 460-3-60	28.8	28.8	28.8	28.8	28.8	29.1	29.1	29.1
575-3-60	28.8	28.8	28.8	28.8	28.8	29.1	29.3	29.3
230-3-50, 400-3-50	23.8	23.8	23.8	23.8	23.8	23.8	23.8	23.8
REFRIGERANT	R-22							
Operating charge (kg) (approx per circuit)[2]	1.36	1.36	0.68/0.68	0.90/0.90	1.13/1.13	1.59/1.59	2.04/2.04	2.27/2.27
DIRECT-EXPANSION COIL	Enhanced Copper Tubes, Aluminum Sine-Wave Fins							
Max Working Pressure (kPag)	2999							
Face Area (sq m)	0.62	0.77	0.93	0.93	1.64	1.85	2.30	2.77
No. of Splits	1	1	2	2	2	2	2	2
No. of Circuits per Split	12	15	9	9	12	13	15	18
Split Type...Percentage	-	-	Face...50/50					
Rows...Fins/m	3...591	3...591	3...670	3...591	3...591	3...670	3...591	3...591
STEAM COIL***								
Max Working Pressure (kPag at 204.4 C)	1207							
Total Face Area (sq m)	0.62	0.62	0.62	1.24	1.24	1.24	1.39	1.39
Rows...Fins/m	1...355	1...355	1...355	1...394	1...394	1...394	1...394	1...394
HOT WATER COIL***								
Max Working Pressure (kPag)	1034							
Total Face Area (sq m)	0.62	0.62	0.62	1.24	1.24	1.24	1.39	1.39
Rows...Fins/m	2...335	2...335	2...335	2...335	2...335	2...335	2...493	2...493
Water Volume								
(L)	31.4			52.6			54.1	
(m^3)	0.031			0.052			0.054	
PIPING CONNECTIONS**								
Quantity...Size (in.)								
DX Coil - Suction (ODF)	1...1 1/8	1...1 1/8	2...1 1/8	2...1 1/8	2...1 1/8	2...1 1/8	2...1 3/8	2...1 3/8
DX Coil - Liquid Refrigerant (ODF)	1...5/8			2...5/8				
Steam Coil, In (MPT)	1...2 1/2			1...2 1/2				
Steam Coil, Out (MPT)	1...1 1/2			1...1 1/2				
Hot Water Coil, In (MPT)	1...1 1/2		1...1 1/2	1...2				
Hot Water Coil, Out (MPT)	1...1 1/2		1...1 1/2	1...2				
Condensate (PVC)	1...1 1/4 ODM/1 IDF							
FILTERS	Washable - Factory Supplied							
Quantity...Size	4...406 x 610 x 25.4			4...406 x 508 x 25.4 4...406 x 610 x 25.4			4...508 x 610 x 25.4 4...508 x 635 x 25.4	
Access Location	Front							

LEGEND

DX - Direct Expansion
TXV - Thermostatic Expansion Valve

*Refer to Alternate Fan Motor Data table, pages 56 and 57, for alternate motor data.
[2]Units are shipped without refrigerant charge.
**All piping sizes are OD inches; equivalent sizes in millimeters follow:
***Please contact your local Carrier sales office for availability.

in.	mm
5/8	15.9
1	25.4
1 1/8	28.6
1 1/4	31.8
1 3/8	34.9
1 1/2	38.1
2	50.8
2 1/8	54.0
2 1/2	63.5

Please contact your local Carrier sales office for the weights of single skin units.

40RM/RMS

5

Physical data (cont)

40RMS - ENGLISH

UNIT 40RMS	008SD	010SD	012SD	014SD	016SD	024SD	028SD	034SD
NOMINAL CAPACITY (Tons)	7 1/2	8 1/2	10	12 1/2	15	20	25	30
OPERATING WEIGHT (lb)								
Base Unit (Double skin)	457	458	458	795	811	817	1203	1210
Plenum	97	97	97	140	140	140	180	180
FANS								
Qty...Diam. (in.)	1...15	1...15	1...15	2...15	2...15	2...15	2...18	2...18
Nominal Airflow (cfm)	3000	3400	4000	5000	6000	8000	10,000	12,000
Airflow Range (cfm)	2250-3750	2250-4250	3000-5000	3750-6250	4500-7500	6000-10,000	7500-12,500	9000-15,000
Nominal Motor Hp (Standard Motor)*								
208/230-1-60	2.4	2.4	-	-	-	-	-	-
208/230-3-60 and 460-3-60	2.4	2.4	2.4	2.9	3.7	5.0	7.5	10.0
575-3-60	2.0	2.0	2.0	3.0	3.0	5.0	7.5	10.0
230-3-50, 400-3-50	2.4	2.4	2.9	2.9	2.9	5.0	7.5	10.0
Motor Speed (rpm)								
208/230-1-60	1725	1725	-	-	-	-	-	-
208/230-3-60 and 460-3-60	1725	1725	1725	1725	1725	1745	1745	1745
575-3-60	1725	1725	1725	1725	1725	1745	1755	1755
230-3-50, 400-3-50	1425							
CHILLED WATER COIL	Enhanced Copper Tubes, Aluminum Sine-Wave Fins							
Max Working Pressure (psig)	435							
Face Area (sq ft) - Upper	8.3	9.0	9.8	8.3	8.3	11.0	12.4	15.5
Face Area (sq ft) - Lower	-	-	-	5.5	8.3	8.3	12.4	12.4
Rows...Fins/in.	3...15							
Water Volume								
(gal)	3.0	3.3	3.5	4.7	5.6	6.4	8.9	9.9
(ft^3)	0.40	0.47	0.46	0.63	0.75	0.85	1.19	1.32
STEAM COIL**								
Max Working Pressure (psig at 400 F)	175							
Total Face Area (sq ft)	6.67	6.67	6.67	13.33	13.33	13.33	15.0	15.0
Rows...Fins/in.	1...9	1...9	1...9	1...10	1...10	1...10	1...10	1...10
HOT WATER COIL**								
Max Working Pressure (in. wg)	150							
Total Face Area (sq ft)	6.67	6.67	6.67	13.33	13.33	13.33	15.0	15.0
Rows...Fins/in.	2...8.5	2...8.5	2...8.5	2...8.5	2...8.5	2...8.5	2...12.5	2...12.5
Water Volume								
(gal)		8.3			13.9		14.3	
(ft^3)		1.1			1.85		1.90	
PIPING CONNECTIONS								
Quantity...Size (in.)								
Chilled Water - In	1...1 3/8 ODF	1...1 3/8 ODF	1...1 3/8 ODF	2...1 3/8 ODM	2...1 3/8 ODM	2...1 3/8 ODM	2...2 1/8 ODM	2...2 1/8 ODM
Chilled Water - Out	1...1 3/8 ODF	1...1 3/8 ODF	1...1 3/8 ODF	2...1 3/8 ODM	2...1 3/8 ODM	2...1 3/8 ODM	2...2 1/8 ODM	2...2 1/8 ODM
Steam Coil, In (MPT)	1...2 1/2				1...2 1/2			
Steam Coil, Out (MPT)	1...1 1/2				1...1 1/2			
Hot Water Coil, In (MPT)	1...1 1/2		1...1 1/2			1...2		
Hot Water Coil, Out (MPT)	1...1 1/2		1...1 1/2			1...2		
Condensate (PVC)	1...1 1/4 ODM/1 IDF							
FILTERS	Washable - Factory Supplied							
Quantity...Size (in.)	4...16 x 24 x 1			4...16 x 20 x 1 4...16 x 24 x 1			4...20 x 24 x 1 4...20 x 25 x 1	
Access Location	Front							

*Refer to the Alternate Fan Motor Data table, pages 52 and 53, for alternate motor data.
**Please contact your local Carrier sales office for availability.

Please contact your local Carrier sales office for the weights of single skin units.

40RM/RMS

14.2. ESP32-C6-Zero - Etapa Controlador

ESP32-C6-Zero

From Waveshare Wiki
Jump to: navigation, search

Overview

The ESP32-C6-Zero is a low-cost, high-performance microcontroller development board with a compact size and rich peripheral interfaces.
Adopts ESP32-C6FH4 as the main chip, with RISC-V 32-bit single-core processor, support up to 160 MHz, and built-in 320KB ROM, 512KB HP SRAM and 16KB LP SRAM. It can be compatible with multiple peripheral devices and is easier to use in different application scenarios.
You can choose ESP-IDF development environment or Arduino IED for development so that you can easily and quickly get started and apply it to the product.

ESP32-C6-Zero
Without header

ESP32-C6, USB Type-C

ESP32-C6-Zero
With pre-soldered header

ESP32-C6, USB Type-C

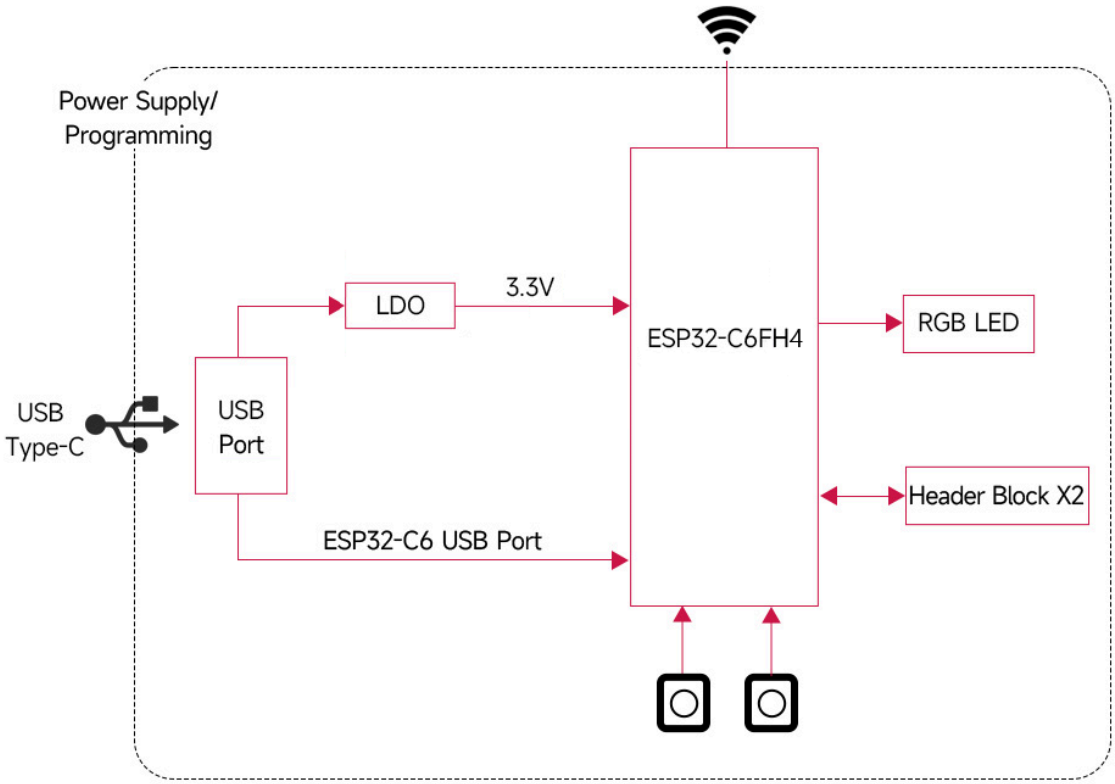
Features

- Adopts ESP32-C6FH4 chip, with RISC-V 32-bit single-core processor, support up to 160 MHz.
- Integrated 320KB ROM, 512KB HP SRAM, 16KB LP SRAM and 4MB Flash memory.
- Integrated WiFi 6, Bluetooth 5 and IEEE 802.15.4 (Zigbee 3.0 and Thread) wireless communication, with superior RF performance.
- Type-C connector, easier to use.
- Rich peripheral interfaces, better compatibility and expandability.
- Castellated module allows soldering directly to carrier boards.

From Waveshare Wiki: Open-source documentation for hundreds of models. For the latest version of the manual or further support, please contact customer service.

- Support multiple low-power modes, adjustable balance between communication distance, data rate and power consumption to meet the power requirements of various application scenarios.

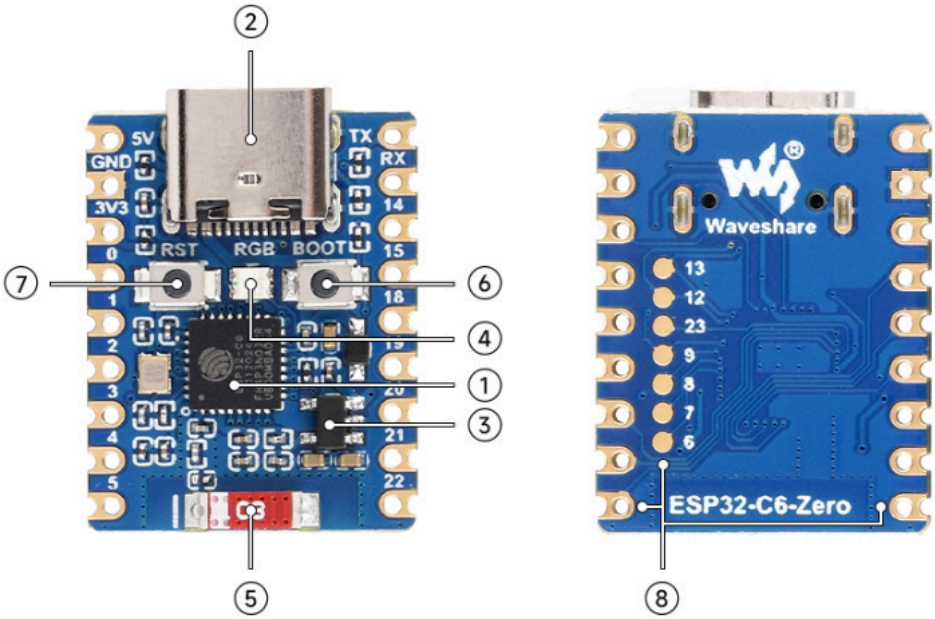
Function Diagram



From Waveshare Wiki: Open-source documentation for hundreds of models. For the latest version of the manual or further support, please contact customer service.

(/wiki/File:ESP32-C6-Zero-Diagram.png)

Onboard Interfaces

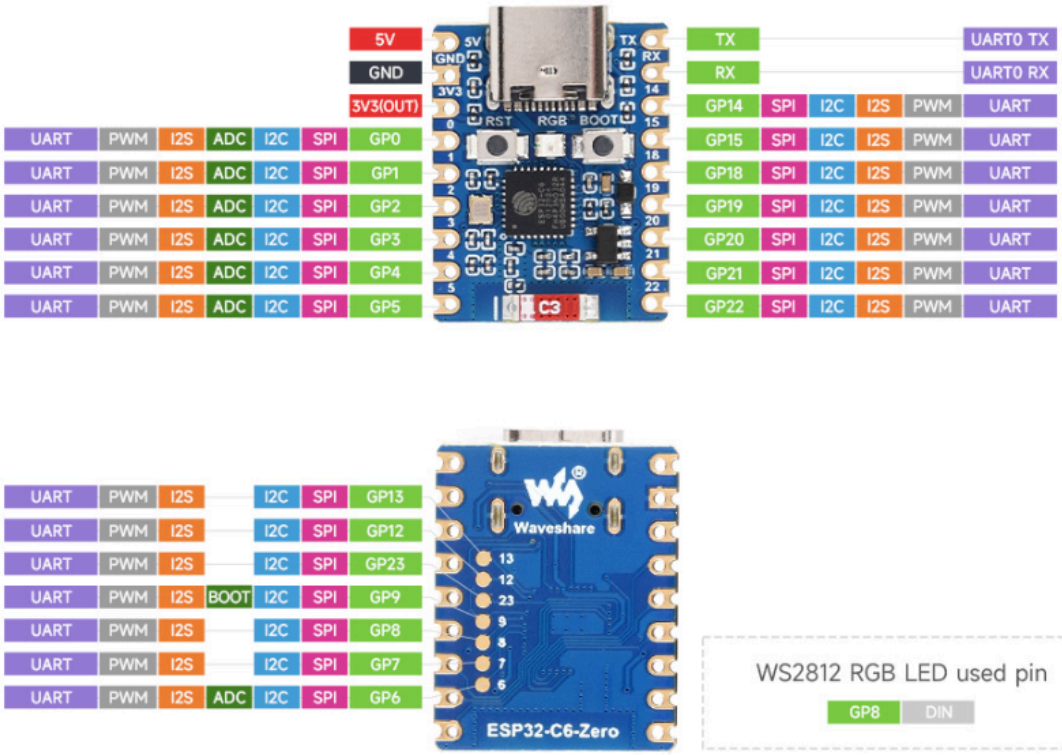


1. **ESP32-C6FH4**
   Built-in dual processors, main frequency is up to 160 MHz
2. **USB Type-C Port**
   for downloading programme and debugging
3. **ME6217C33M5G**
   low dropout LDO, 800mA (Max)
4. **WS2812 RGB LED**
5. **2.4G ceramic antenna**
6. **BOOT button**
   Press it and then press the RESET button to enter download mode
7. **RESET button**
8. **ESP32-C6FH4 pins**

From Waveshare Wiki: Open-source documentation for hundreds of models. For the latest version of the manual or further support, please contact customer service.

(/wiki/File:ESP32-C6-Zero_Inter.png)

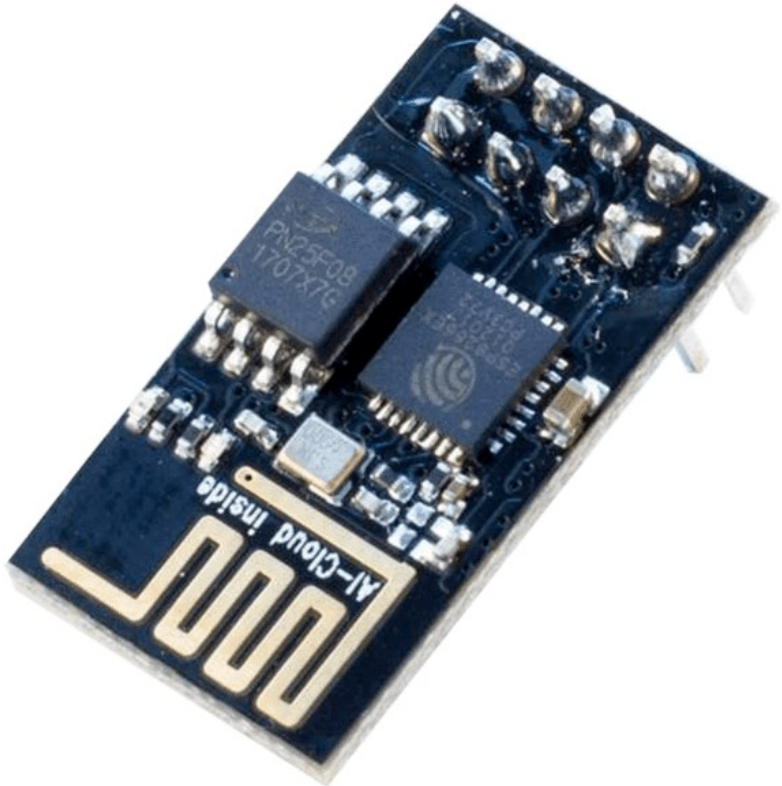
Pinout



From Waveshare Wiki: Open-source documentation for hundreds of models. For the latest version of the manual or further support, please contact customer service.

14.3. ESP-01S - Etapa Sensado

ESP8266-01S Modul Datenblatt



ESP8266-01S Modul Datenblatt

Contents:

ESP8266-01S Modul Datenblatt

1. Description

ESP-01 WiFi module is developed by Ai-thinker Team. Core processor ESP8266 in smaller sizes of the module encapsulates Tensilica L106 integrates ultra low power 32-bit bit MCU micro, with the 16 16-bit short mode. Clock speed support 80MHz, 160MHz, supports the RTOS, integrated Wi-Fi MAC/BB/ RF/PA/LNA, /BB/RF/PA/LNA, on on-board antenna.

The module supports standard IEEE802.11 b/g/n agreement, complete TCP/IP protocol stack. Users can use the add modules to an existing device networking, or building a separate network controller.

ESP8266 is high integration wireless SOCs, designed for space and power. It provides unsurpassed ability to embed Wi-Fi capabilities within other systems, or to function as a standalone application, with the lowest cost, and minimal space requirement.

ESP8266EX offers a complete and self-contained Wi-Fi networking solution. It can be used to host the application or to offload Wi-Fi networking functions from another application processor.

When ESP8266EX hosts the application, it boots up directly from an external flash. In has integrated cache to improve the performance of the system in such applications.

Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any microcontroller based design with simple connectivity (SPI/SDIO or I2C/ UART interface).

ESP8266EX is among the most integrated WiFi chip in the industry. It integrates the antenna switches, RF balun, power amplifier, low noise receive amplifier, filters, power management modules. It requires minimal external circuitry, and the entire solution, including front-end module. It is designed to occupy minimal PCB area.

ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor, with on-chip SRAM, besides the Wi-Fi functionalities. ESP8266EX is often integrated with external sensors and other application specific devices through its GPIOs. Codes for such applications are provided in the examples in the SDK.

Espressif Systems' Smart Connectivity Platform (ESCP) demonstrates sophisticated system level features include fast sleep/wake context switching for energy-efficient VoIP, adaptive radio biasing, for low-power operation, advanced signal processing, and spur cancellation and radio co-existence features for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation.

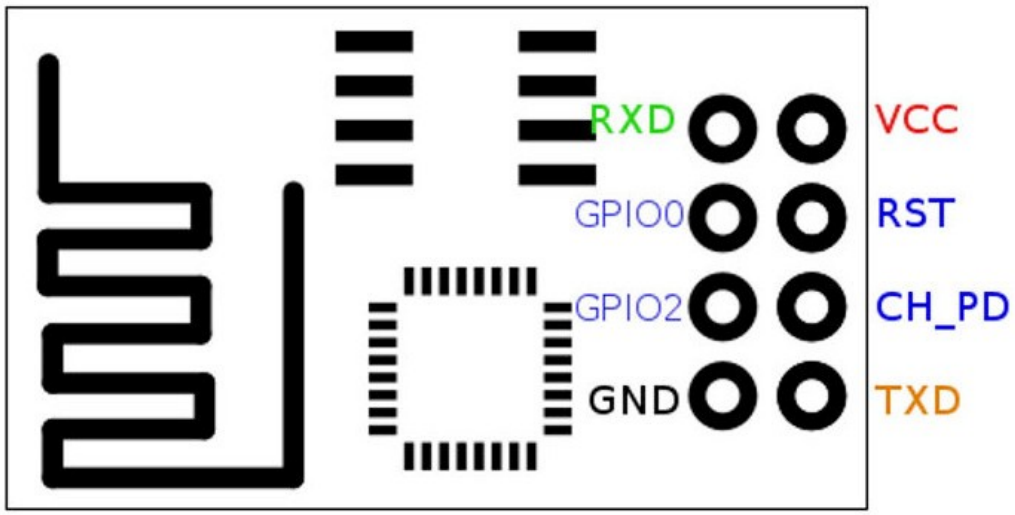
AZ-Delivery

ESP8266-01S Modul Datenblatt

2. Specifications

- 802.11 b/g/n
- Integrated low power 32-bit MCU
- Integrated 10-bit ADC
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLL, regulators, and power management units
- Supports antenna diversity
- Wi-Fi 2.4 GHz, support WPA/WPA2
- Support STA/AP/STA+AP operation modes
- Support Smart Link Function for both Android and iOS devices
- Support Smart Link Function for both Android and iOS devices
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM, GPIO
- STBC, 1x1 MIMO, 2x1 MIMO
- A-MPDU & A-MSDU aggregation and 0.4s guard interval
- Deep sleep power <10uA, Power down leakage current < 5uA
- Wake up and transmit packets in <2ms
- Standby power consumption of <1.0mW (DTIM3)
- +20dBm output power in 802.11b mode
- Operating temperature range -40°C ~ 125°C

3. Pin Configuration



VCC	+3.3V power supply
GND	Ground (0V)
GPIO0	General Purpose Input/Output pin 0
GPIO2	General Purpose Input/Output pin 2
CH_PD	Chip Enable
RST	Reset
RX	Receive line of Serial Interface
TX	Transmit line of Serial Interface

14.4. Convertidor Step-Down LM2596

TEXAS INSTRUMENTS

LM2596

www.ti.com SNVS124C – NOVEMBER 1999 – REVISED APRIL 2013

LM2596 SIMPLE SWITCHER® Power Converter 150 kHz 3A Step-Down Voltage Regulator

Check for Samples: LM2596

FEATURES

- **3.3V, 5V, 12V, and Adjustable Output Versions**
- **Adjustable Version Output Voltage Range, 1.2V to 37V ±4% Max Over Line and Load Conditions**
- **Available in TO-220 and TO-263 Packages**
- **Ensured 3A Output Load Current**
- **Input Voltage Range Up to 40V**
- **Requires Only 4 External Components**
- **Excellent Line and Load Regulation Specifications**
- **150 kHz Fixed Frequency Internal Oscillator**
- **TTL Shutdown Capability**
- **Low Power Standby Mode, I_Q Typically 80 µA**
- **High Efficiency**
- **Uses Readily Available Standard Inductors**
- **Thermal Shutdown and Current Limit Protection**

APPLICATIONS

- **Simple High-Efficiency Step-Down (Buck) Regulator**
- **On-Card Switching Regulators**
- **Positive to Negative Converter**

DESCRIPTION

The LM2596 series of regulators are monolithic integrated circuits that provide all the active functions for a step-down (buck) switching regulator, capable of driving a 3A load with excellent line and load regulation. These devices are available in fixed output voltages of 3.3V, 5V, 12V, and an adjustable output version.

Requiring a minimum number of external components, these regulators are simple to use and include internal frequency compensation , and a fixed-frequency oscillator.

The LM2596 series operates at a switching frequency of 150 kHz thus allowing smaller sized filter components than what would be needed with lower frequency switching regulators. Available in a standard 5-lead TO-220 package with several different lead bend options, and a 5-lead TO-263 surface mount package.

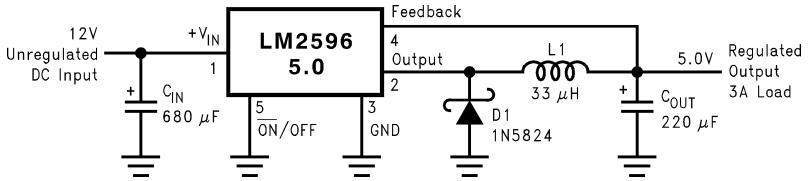
A standard series of inductors are available from several different manufacturers optimized for use with the LM2596 series. This feature greatly simplifies the design of switch-mode power supplies.

Other features include a ensured ±4% tolerance on output voltage under specified input voltage and output load conditions, and ±15% on the oscillator frequency. External shutdown is included, featuring typically 80 µA standby current. Self protection features include a two stage frequency reducing current limit for the output switch and an over temperature shutdown for complete protection under fault conditions. (1)

(1) † Patent Number 5,382,918.

Typical Application

(Fixed Output Voltage Versions)



Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

SIMPLE SWITCHER is a registered trademark of Texas Instruments.
All other trademarks are the property of their respective owners.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of the Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

Connection Diagrams

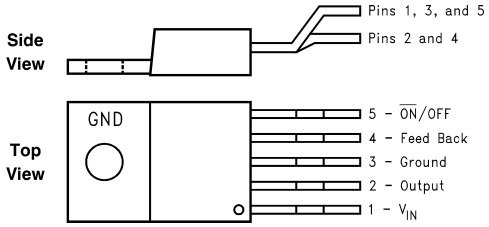


Figure 1. 5-Lead Bent and Staggered Leads, Through Hole TO-220 (T) Package See Package Number NDH0005D

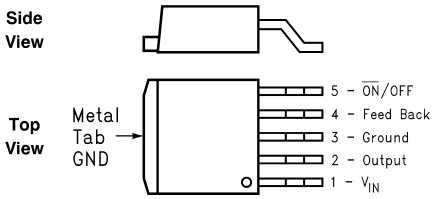


Figure 2. 5-Lead DDPAK/TO-263 (S) Package See Package Number KTT0005B

These devices have limited built-in ESD protection. The leads should be shorted together or the device placed in conductive foam during storage or handling to prevent electrostatic damage to the MOS gates.

Absolute Maximum Ratings (1)(2)

Maximum Supply Voltage	45V
$\overline{ON}$ /OFF Pin Input Voltage	−0.3 ≤ V ≤ +25V
Feedback Pin Voltage	−0.3 ≤ V ≤+25V
Output Voltage to Ground (Steady State)	−1V
Power Dissipation	Internally limited
Storage Temperature Range	−65°C to +150°C
ESD Susceptibility	
Human Body Model (3)	2 kV
Lead Temperature	
DDPAK/TO-263 Package	
Vapor Phase (60 sec.)	+215°C
Infrared (10 sec.)	+245°C
TO-220 Package (Soldering, 10 sec.)	+260°C
Maximum Junction Temperature	+150°C

(1) Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. Operating Ratings indicate conditions for which the device is intended to be functional, but do not ensure specific performance limits. For ensured specifications and test conditions, see the Electrical Characteristics.
(2) If Military/Aerospace specified devices are required, please contact the Texas Instruments Sales Office/ Distributors for availability and specifications.
(3) The human body model is a 100 pF capacitor discharged through a 1.5k resistor into each pin.

Operating Conditions

Temperature Range	−40°C ≤ T_J ≤ +125°C
Supply Voltage	4.5V to 40V

LM2596-3.3 Electrical Characteristics

Specifications with standard type face are for T_J = 25°C, and those with **boldface type** apply over **full Operating Temperature Range**

Symbol	Parameter	Conditions	LM2596-3.3		Units (Limits)
			Typ (1)	Limit (2)	
SYSTEM PARAMETERS (3) Test Circuit Figure 20					
V_{OUT}	Output Voltage	4.75V ≤ V_{IN} ≤ 40V, 0.2A ≤ I_{LOAD} ≤ 3A	3.3		V
				3.168/**3.135**	V(min)
				3.432/**3.465**	V(max)
η	Efficiency	V_{IN} = 12V, I_{LOAD} = 3A	73		%

(1) Typical numbers are at 25°C and represent the most likely norm.
(2) All limits specified at room temperature (standard type face) and at temperature extremes (bold type face). All room temperature limits are 100% production tested. All limits at temperature extremes are ensured via correlation using standard Statistical Quality Control (SQC) methods. All limits are used to calculate Average Outgoing Quality Level (AOQL).
(3) External components such as the catch diode, inductor, input and output capacitors, and voltage programming resistors can affect switching regulator system performance. When the LM2596 is used as shown in the Figure 20 test circuit, system performance will be as shown in system parameters of Electrical Characteristics section.

LM2596-5.0 Electrical Characteristics

Specifications with standard type face are for T_J = 25°C, and those with **boldface type** apply over **full Operating Temperature Range**

Symbol	Parameter	Conditions	LM2596-5.0		Units (Limits)
			Typ (1)	Limit (2)	
SYSTEM PARAMETERS (3) Test Circuit Figure 20					
V_{OUT}	Output Voltage	7V ≤ V_{IN} ≤ 40V, 0.2A ≤ I_{LOAD} ≤ 3A	5.0		V
				4.800/**4.750**	V(min)
				5.200/**5.250**	V(max)
η	Efficiency	V_{IN} = 12V, I_{LOAD} = 3A	80		%

(1) Typical numbers are at 25°C and represent the most likely norm.
(2) All limits specified at room temperature (standard type face) and at temperature extremes (bold type face). All room temperature limits are 100% production tested. All limits at temperature extremes are ensured via correlation using standard Statistical Quality Control (SQC) methods. All limits are used to calculate Average Outgoing Quality Level (AOQL).
(3) External components such as the catch diode, inductor, input and output capacitors, and voltage programming resistors can affect switching regulator system performance. When the LM2596 is used as shown in the Figure 20 test circuit, system performance will be as shown in system parameters of Electrical Characteristics section.

LM2596-12 Electrical Characteristics

Specifications with standard type face are for T_J = 25°C, and those with **boldface type** apply over **full Operating Temperature Range**

Symbol	Parameter	Conditions	LM2596-12		Units (Limits)
			Typ (1)	Limit (2)	
SYSTEM PARAMETERS (3) Test Circuit Figure 20					
V_{OUT}	Output Voltage	15V ≤ V_{IN} ≤ 40V, 0.2A ≤ I_{LOAD} ≤ 3A	12.0		V
				11.52/**11.40**	V(min)
				12.48/**12.60**	V(max)
η	Efficiency	V_{IN} = 25V, I_{LOAD} = 3A	90		%

(1) Typical numbers are at 25°C and represent the most likely norm.
(2) All limits specified at room temperature (standard type face) and at temperature extremes (bold type face). All room temperature limits are 100% production tested. All limits at temperature extremes are ensured via correlation using standard Statistical Quality Control (SQC) methods. All limits are used to calculate Average Outgoing Quality Level (AOQL).
(3) External components such as the catch diode, inductor, input and output capacitors, and voltage programming resistors can affect switching regulator system performance. When the LM2596 is used as shown in the Figure 20 test circuit, system performance will be as shown in system parameters of Electrical Characteristics section.

LM2596-ADJ Electrical Characteristics

Specifications with standard type face are for T_J = 25°C, and those with **boldface type** apply over **full Operating Temperature Range**

Symbol	Parameter	Conditions	LM2596-ADJ		Units (Limits)
			Typ (1)	Limit (2)	
SYSTEM PARAMETERS (3) Test Circuit Figure 20					
V_{FB}	Feedback Voltage	4.5V ≤ V_{IN} ≤ 40V, 0.2A ≤ I_{LOAD} ≤ 3A V_{OUT} programmed for 3V. Circuit of Figure 20	1.230	 1.193/**1.180** 1.267/**1.280**	V V(min) V(max)
η	Efficiency	V_{IN} = 12V, V_{OUT} = 3V, I_{LOAD} = 3A	73		%

(1) Typical numbers are at 25°C and represent the most likely norm.
(2) All limits specified at room temperature (standard type face) and at temperature extremes (bold type face). All room temperature limits are 100% production tested. All limits at temperature extremes are ensured via correlation using standard Statistical Quality Control (SQC) methods. All limits are used to calculate Average Outgoing Quality Level (AOQL).
(3) External components such as the catch diode, inductor, input and output capacitors, and voltage programming resistors can affect switching regulator system performance. When the LM2596 is used as shown in the Figure 20 test circuit, system performance will be as shown in system parameters of Electrical Characteristics section.

All Output Voltage Versions Electrical Characteristics

Specifications with standard type face are for T_J = 25°C, and those with **boldface type** apply over **full Operating Temperature Range**. Unless otherwise specified, V_{IN} = 12V for the 3.3V, 5V, and Adjustable version and V_{IN} = 24V for the 12V version. I_{LOAD} = 500 mA

Symbol	Parameter	Conditions	LM2596-XX		Units (Limits)
			Typ (1)	Limit (2)	
DEVICE PARAMETERS					
I_b	Feedback Bias Current	Adjustable Version Only, V_{FB} = 1.3V	10	 50/**100**	nA nA (max)
f_O	Oscillator Frequency	See (3)	150	 127/**110** 173/**173**	kHz kHz(min) kHz(max)
V_{SAT}	Saturation Voltage	I_{OUT} = 3A (4) (5)	1.16	 1.4/**1.5**	V V(max)
DC	Max Duty Cycle (ON)	See (5)	100		%
	Min Duty Cycle (OFF)	See (6)	0		
I_{CL}	Current Limit	Peak Current (4)(5)	4.5	 3.6/**3.4** 6.9/**7.5**	A A(min) A(max)
I_L	Output Leakage Current	Output = 0V (4)(6)		50	µA(max)
		Output = −1V (7)	2	 30	mA mA(max)
I_Q	Quiescent Current	See (6)	5	 10	mA mA(max)

(1) Typical numbers are at 25°C and represent the most likely norm.
(2) All limits specified at room temperature (standard type face) and at temperature extremes (bold type face). All room temperature limits are 100% production tested. All limits at temperature extremes are ensured via correlation using standard Statistical Quality Control (SQC) methods. All limits are used to calculate Average Outgoing Quality Level (AOQL).
(3) The switching frequency is reduced when the second stage current limit is activated.
(4) No diode, inductor or capacitor connected to output pin.
(5) Feedback pin removed from output and connected to 0V to force the output transistor switch ON.
(6) Feedback pin removed from output and connected to 12V for the 3.3V, 5V, and the ADJ. version, and 15V for the 12V version, to force the output transistor switch OFF.
(7) V_{IN} = 40V.

All Output Voltage Versions Electrical Characteristics (continued)

Specifications with standard type face are for T_J = 25°C, and those with **boldface type** apply over **full Operating Temperature Range**. Unless otherwise specified, V_{IN} = 12V for the 3.3V, 5V, and Adjustable version and V_{IN} = 24V for the 12V version. I_{LOAD} = 500 mA

Symbol	Parameter	Conditions	LM2596-XX		Units (Limits)
			Typ (1)	Limit (2)	
I_{STBY}	Standby Quiescent Current	ON/OFF pin = 5V (OFF) (7)	80		μA
				200/**250**	μA(max)
θ_{JC}	Thermal Resistance	TO-220 or TO-263 Package, Junction to Case	2		°C/W
θ_{JA}		TO-220 Package, Junction to Ambient (8)	50		°C/W
θ_{JA}		TO-263 Package, Junction to Ambient (9)	50		°C/W
θ_{JA}		TO-263 Package, Junction to Ambient (10)	30		°C/W
θ_{JA}		TO-263 Package, Junction to Ambient (11)	20		°C/W
ON/OFF CONTROL Test Circuit Figure 20					
	$\overline{ON}$ /OFF Pin Logic Input		1.3		V
V_{IH}	Threshold Voltage	Low (Regulator ON)		**0.6**	V(max)
V_{IL}		High (Regulator OFF)		**2.0**	V(min)
I_H	$\overline{ON}$ /OFF Pin Input Current	V_{LOGIC} = 2.5V (Regulator OFF)	5		μA
				15	μA(max)
I_L		V_{LOGIC} = 0.5V (Regulator ON)	0.02		μA
				5	μA(max)

(8) Junction to ambient thermal resistance (no external heat sink) for the TO-220 package mounted vertically, with the leads soldered to a printed circuit board with (1 oz.) copper area of approximately 1 in^2.
(9) Junction to ambient thermal resistance with the TO-263 package tab soldered to a single printed circuit board with 0.5 in^2 of (1 oz.) copper area.
(10) Junction to ambient thermal resistance with the TO-263 package tab soldered to a single sided printed circuit board with 2.5 in^2 of (1 oz.) copper area.
(11) Junction to ambient thermal resistance with the TO-263 package tab soldered to a double sided printed circuit board with 3 in^2 of (1 oz.) copper area on the LM2596S side of the board, and approximately 16 in^2 of copper on the other side of the p-c board. See Application Information in this data sheet and the thermal model in Switchers Made Simple™ version 4.3 software.

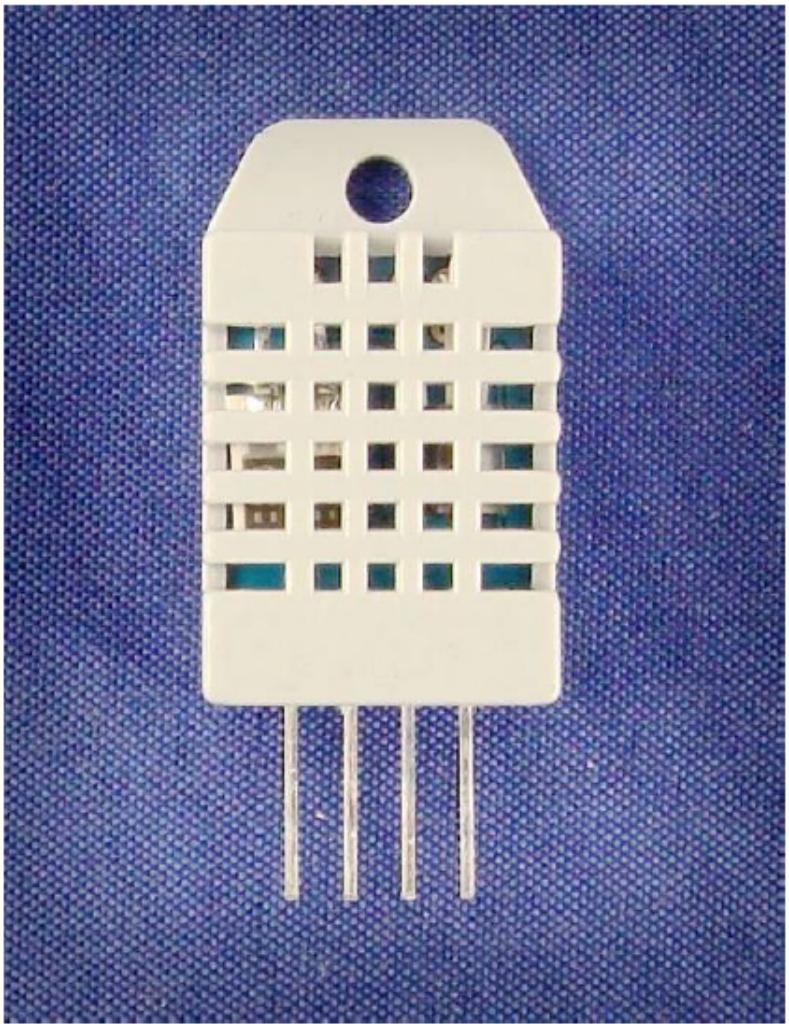
14.5. Sensor DHT22

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

Digital-output relative humidity & temperature sensor/module

DHT22 (DHT22 also named as AM2302)



Capacitive-type humidity and temperature module/sensor

1

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

1. Feature & Application:

* Full range temperature compensated * Relative humidity and temperature measurement
* Calibrated digital signal *Outstanding long-term stability *Extra components not needed
* Long transmission distance * Low power consumption *4 pins packaged and fully interchangeable

2. Description:

DHT22 output calibrated digital signal. It utilizes exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability.Its sensing elements is connected with 8-bit single-chip computer.

Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory.

Small size & low consumption & long transmission distance(20m) enable DHT22 to be suited in all kinds of harsh application occasions.

Single-row packaged with four pins, making the connection very convenient.

3. Technical Specification:

Model	DHT22
Power supply	3.3-6V DC
Output signal	digital signal via single-bus
Sensing element	Polymer capacitor
Operating range	humidity 0-100%RH; temperature -40~80Celsius
Accuracy	humidity +-2%RH(Max +-5%RH); temperature <+-0.5Celsius
Resolution or sensitivity	humidity 0.1%RH; temperature 0.1Celsius
Repeatability	humidity +-1%RH; temperature +-0.2Celsius
Humidity hysteresis	+-0.3%RH
Long-term Stability	+-0.5%RH/year
Sensing period	Average: 2s
Interchangeability	fully interchangeable
Dimensions	small size 14*18*5.5mm; big size 22*28*5mm

4. Dimensions: (unit----mm)

1) Small size dimensions: (unit----mm)

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

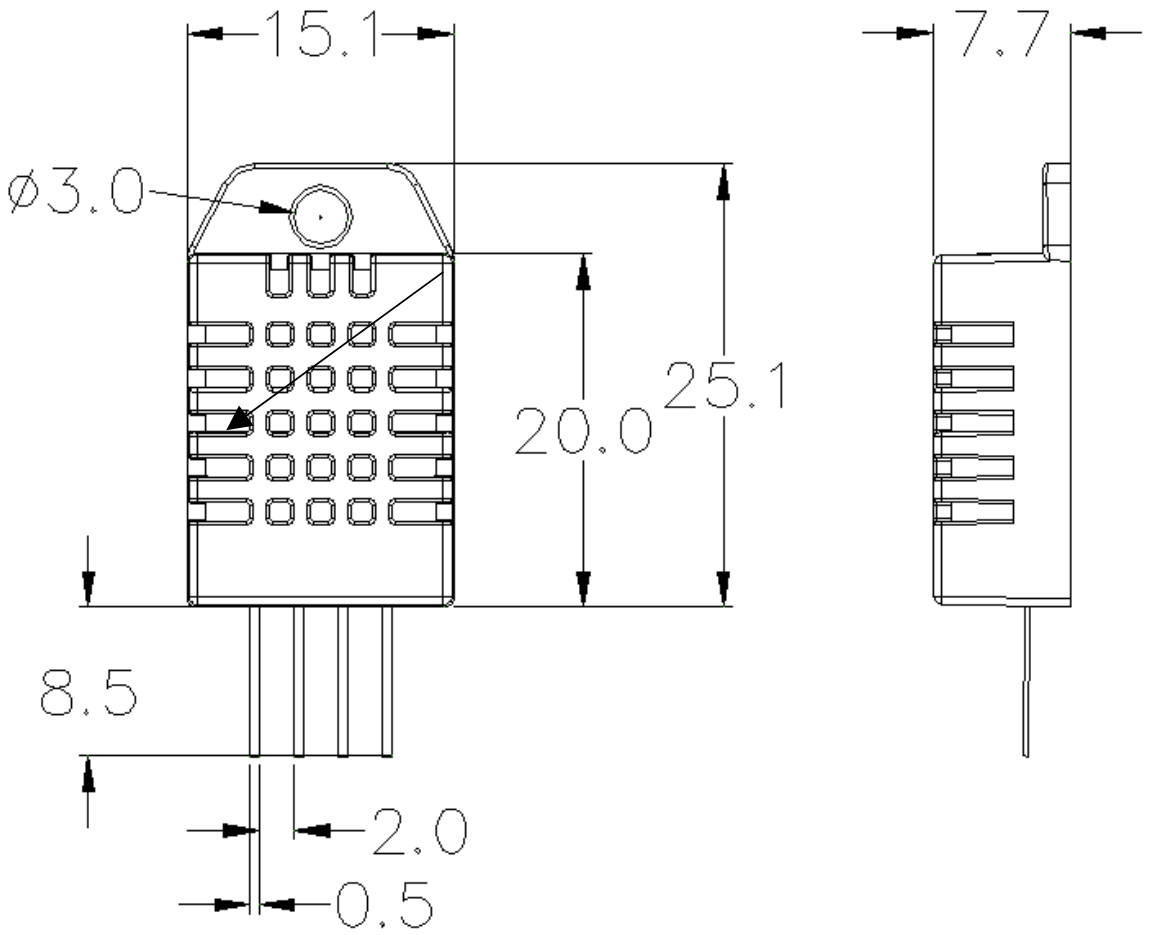
3

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors



Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD----power supply
2	DATA--signal
3	NULL
4	GND

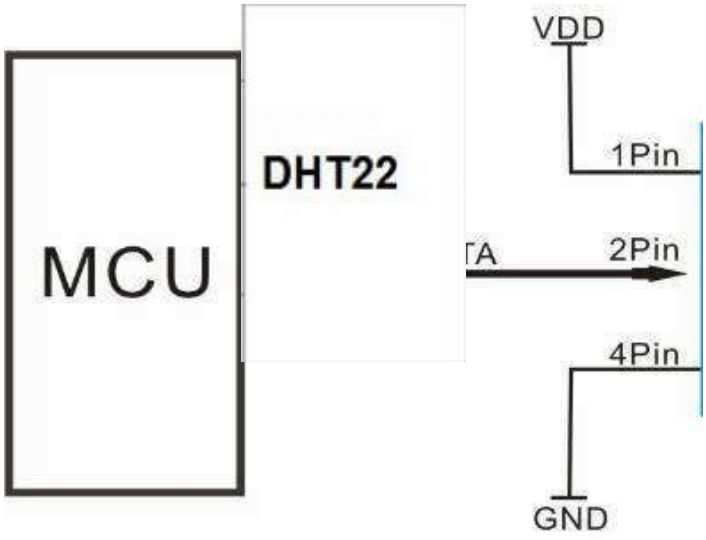
Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

5. Electrical connection diagram:



3Pin---NC, AM2302 is another name for DHT22

6. Operating specifications:

(1) Power and Pins

Power's voltage should be 3.3-6V DC. When power is supplied to sensor, don't send any instruction to the sensor within one second to pass unstable status. One capacitor valued 100nF can be added between VDD and GND for wave filtering.

(2) Communication and signal

Single-bus data is used for communication between MCU and DHT22, it costs 5mS for single time communication.

Data is comprised of integral and decimal part, the following is the formula for data.

DHT22 send out higher data bit firstly!

DATA=8 bit integral RH data+8 bit decimal RH data+8 bit integral T data+8 bit decimal T data+8 bit check-sum

If the data transmission is right, check-sum should be the last 8 bit of "8 bit integral RH data+8 bit decimal RH data+8 bit integral T data+8 bit decimal T data".

When MCU send start signal, DHT22 change from low-power-consumption-mode to running-mode. When MCU finishs sending the start signal, DHT22 will send response signal of 40-bit data that reflect the relative humidity

Thomas Liu (Business Manager)

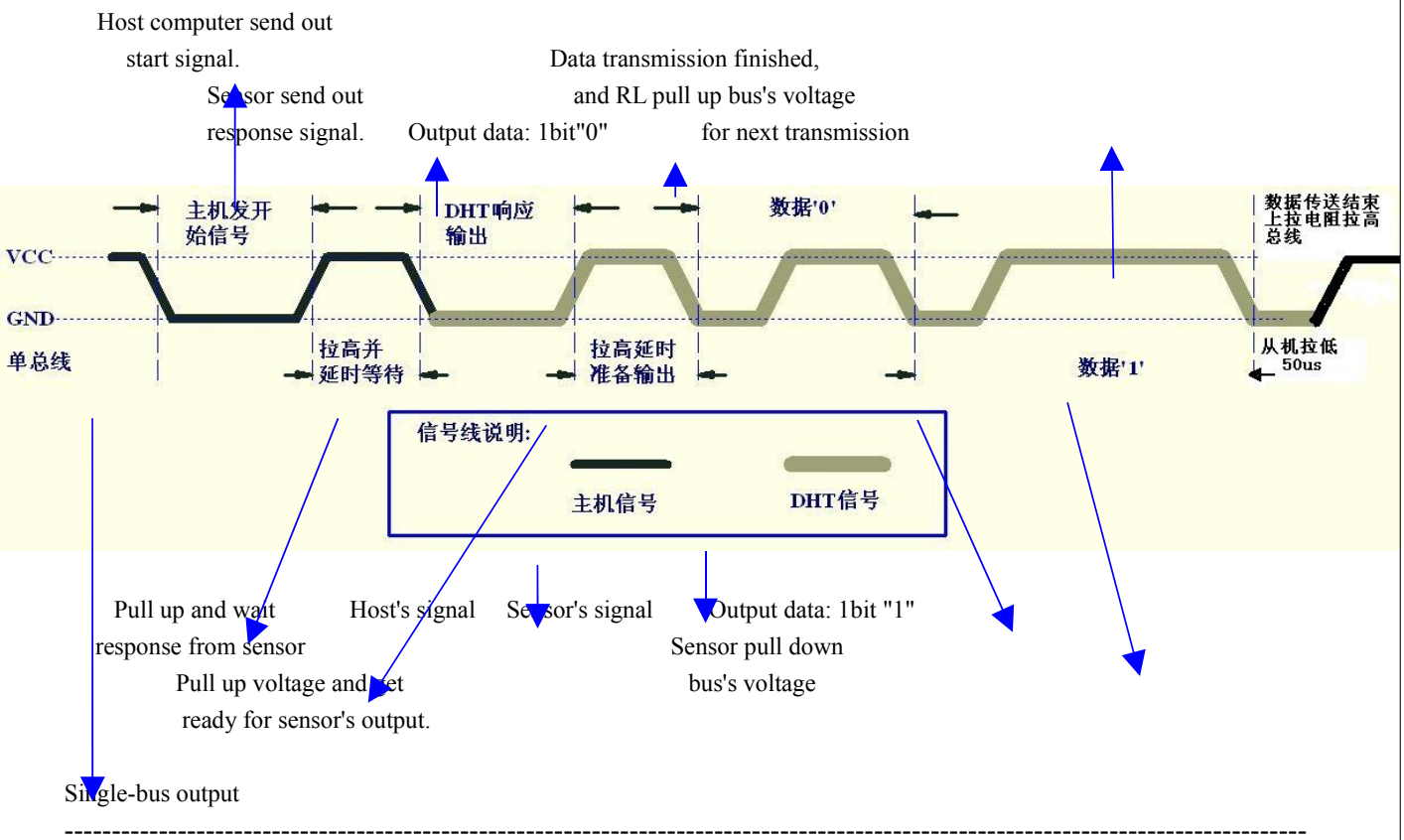
Email: thomasliu198518@yahoo.com.cn

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

and temperature information to MCU. Without start signal from MCU, DHT22 will not give response signal to MCU. One start signal for one time's response data that reflect the relative humidity and temperature information from DHT22. DHT22 will change to low-power-consumption-mode when data collecting finish if it don't receive start signal from MCU again.

1) Check bellow picture for overall communication process:



---

2) Step 1: MCU send out start signal to DHT22

Data-bus's free status is high voltage level. When communication between MCU and DHT22 begin, program of MCU will transform data-bus's voltage level from high to low level and this process must beyond at least 1ms to ensure DHT22 could detect MCU's signal, then MCU will wait 20-40us for DHT22's response.

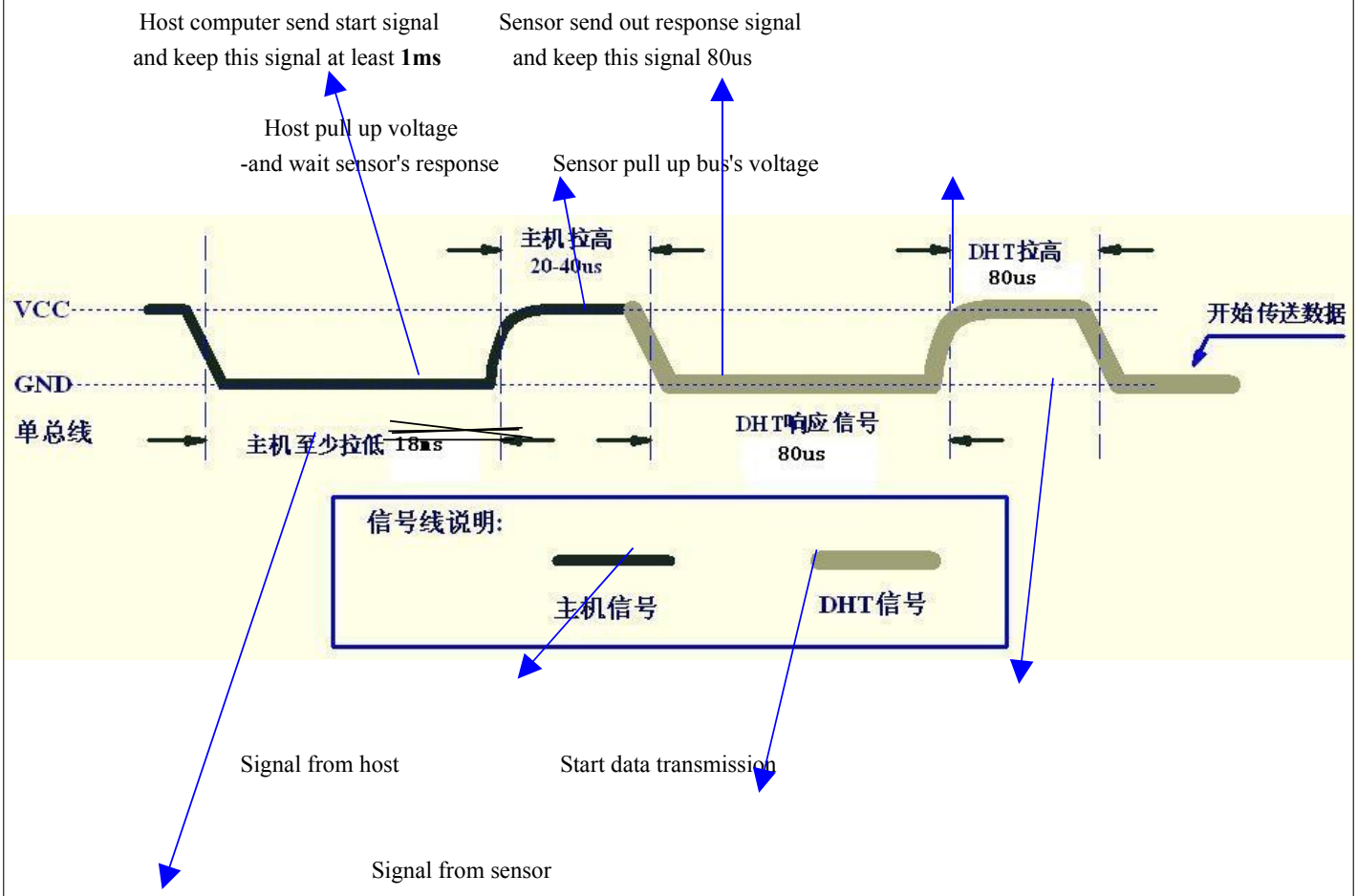
Check bellow picture for step 1:

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors



Single-bus signal

Step 2: DHT22 send response signal to MCU

When DHT22 detect the start signal, DHT22 will send out low-voltage-level signal and this signal last 80us as response signal, then program of DHT22 transform data-bus's voltage level from low to high level and last 80us for DHT22's preparation to send data.

Check bellow picture for step 2:

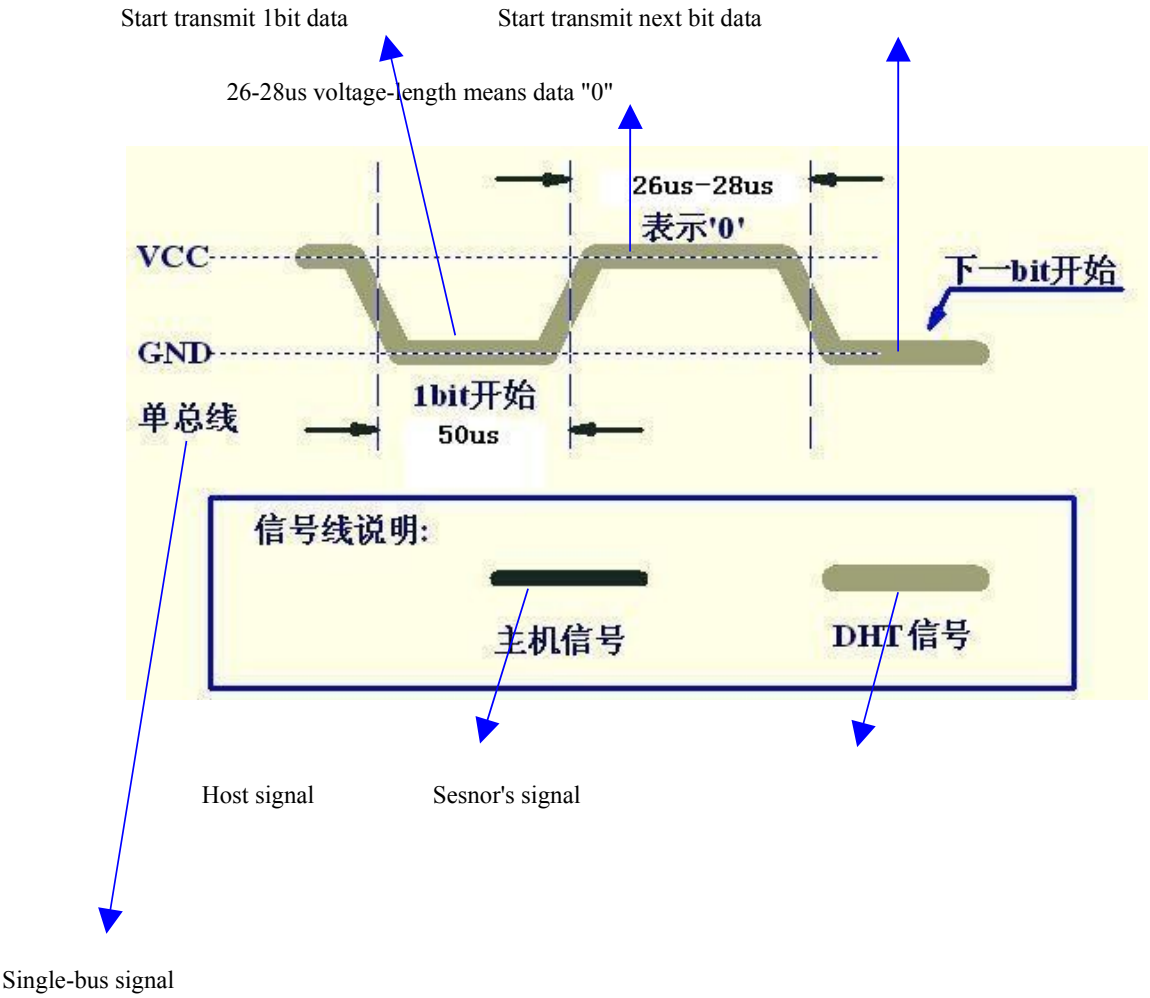
7

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors



Step 3: DHT22 send data to MCU

When DHT22 is sending data to MCU, every bit's transmission begin with low-voltage-level that last 50us, the following high-voltage-level signal's length decide the bit is "1" or "0".

Check bellow picture for step 3:

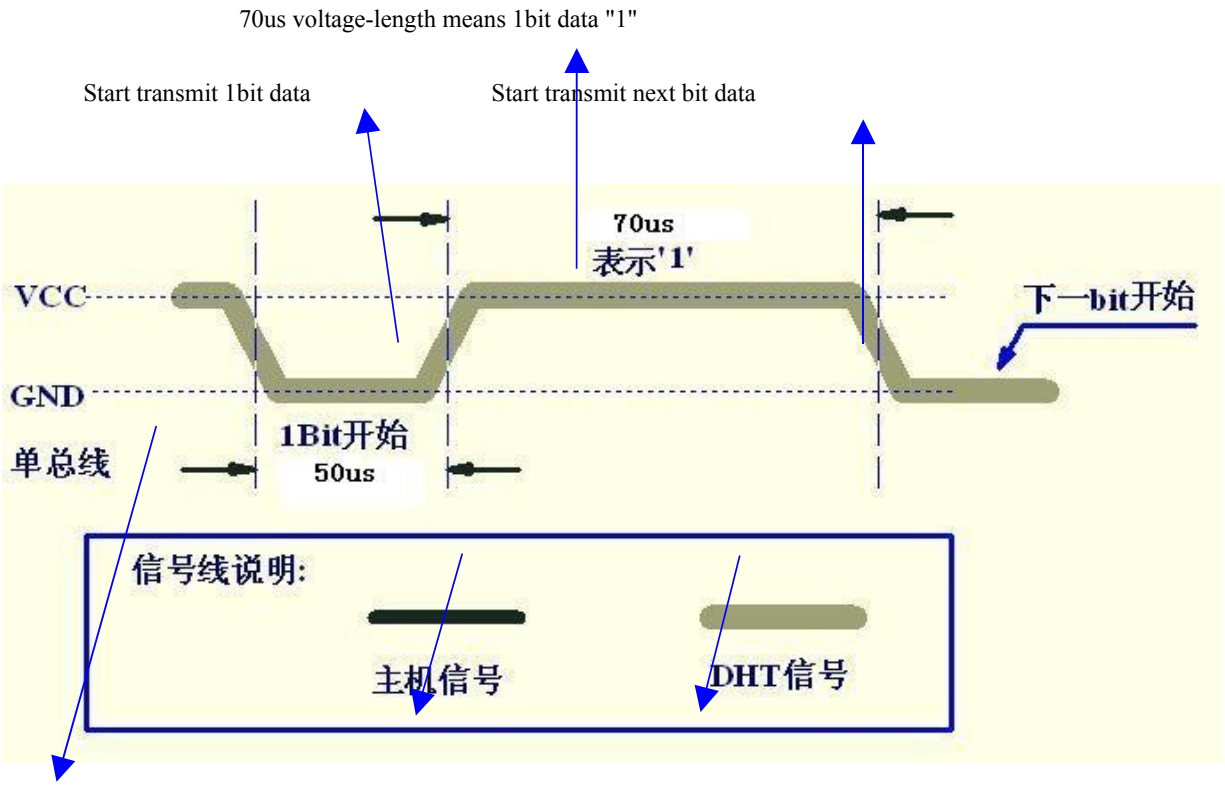
8

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors



Host signal Sesnor's signal

Single-bus signal

If signal from DHT22 is always high-voltage-level, it means DHT22 is not working properly, please check the electrical connection status.

7. Electrical Characteristics:

Item	Condition	Min	Typical	Max	Unit
Power supply	DC	3.3	5	6	V
Current supply	Measuring	1		1.5	mA
	Stand-by	40	Null	50	uA
Collecting period	Second		2		Second

*Collecting period should be : >2 second.

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Referencias

[1] World Health Organization, *WHO Housing and health guidelines*, ISBN: 978-92-4-155037-6, World Health Organization, Ginebra, 2018. visitado 15 de feb. de 2024. dirección: https://www.who.int/publications/i/item/9789241550376.

[2] Y. A. Cengel y M. A. Boles, *Termodinámica*, 2.ª ed. México: McGraw Hill, 2012.

[3] R. Karwa, *Heat and Mass Transfer*. Springer Singapore, 2016.

[4] K. Ogata, *Modern Control Engineering*, 5.ª ed. Boston: Pearson, 2010.

[5] J. Normey-Rico y E. Camacho, *Control of Dead-Time Processes*. Springer, 2007.

[6] G. C. Goodwin, S. F. Graebe y M. E. Salgado, *Control System Design*. New Jersey: Prentice Hall, 2000.

[7] MQTT.org. «MQTT: An Introduction,» visitado 1 de jun. de 2024. dirección: https://mqtt.org/.

[8] IBM, *Mensajería de MQTT*, IBM Cloud Education, 2023. visitado 20 de mayo de 2024. dirección: https://www.ibm.com/topics/mqtt.

[9] I. Petrovsky, *Lecciones sobre ecuaciones en derivadas parciales*. Interscience New York, 1954.

[10] J. Stewart, *Cálculo: Trascendentes tempranas*, 4.ª ed. Thomson Learning, 1999.

[11] M. Balanzat, *Matemática Avanzada para la Física*. Eudeba Manuales, 1994.

[12] G. B. Thomas, *CÁLCULO, varias variables*, 12.ª ed. Pearson, 2012.

[13] D. G. Zill, W. S. Wright y J. Ibarra, *Matemáticas 3: Cálculo de varias variables*. McGraw Hill, 2011.

[14] Universidad Nacional de Misiones, *Sistemas de transmisión: engranajes*, Apunte de Cátedra, Facultad de Ingeniería - UNaM, 2018. dirección: https://www.fio.unam.edu.ar/.

[15] H. Salazar y R. Rossi, *Construcción de engranajes*, Apunte de Cátedra, Universidad Nacional de Rosario (UNR), 2007.

[16] MathWorks. «What Is Hardware-in-the-Loop (HIL)?» Visitado 20 de mar. de 2024. dirección: https://www.mathworks.com/discovery/hardware-in-the-loop-hil.html.