UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE GUAYAQUIL

CARRERA DE INGENIERÍA ELECTRÓNICA

Prototipo de incubación artificial para huevos de codorniz utilizando control difuso y tecnología IOT

Trabajo de titulación previo a la obtención del

Título de Ingeniero en Electrónica

AUTORES: Carlos Aurelio Ordóñez Urgiles

Jerry Bolívar Rivera Murillo

TUTOR: Ing. Rafael Enríquez Pérez Ordóñez MSc.

Guayaquil – Ecuador

2024

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Carlos Aurelio Ordóñez Urgiles con documento de identificación N° 0953343373 y Jerry Bolívar Rivera Murillo con documento de identificación N° 0954621033, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del proyecto de investigación: "Prototipo de incubación artificial para huevos de codorniz utilizando control difuso y tecnología IOT", el cual ha sido desarrollado para optar por el título de: Ingeniero Electrónico, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Guayaquil, 11 de marzo de 2024

Atentamente,

Carlos Aurelio Ordóñez Urgiles
0953343373

Jerry Bolívar Rivera Murillo
0954621033

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Carlos Aurelio Ordóñez Urgiles con documento de identificación N° 0953343373 y Jerry Bolívar Rivera Murillo con documento de identificación N° 0954621033; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Guayaquil, 11 de marzo de 2024

Atentamente,

Carlos Aurelio Ordóñez Urgiles
0953343373

Jerry Bolívar Rivera Murillo
0954621033

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Rafael Enrique Pérez Ordoñez con documento de identificación N° 0916275076, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: "Prototipo de incubación artificial para huevos de codorniz utilizando control difuso y tecnología IOT" realizado Carlos Aurelio Ordóñez Urgiles con documento de identificación N° 0953343373 y Jerry Bolívar Rivera Murillo con documento de identificación N° 0954621033, obteniendo como resultado final el trabajo de titulación bajo la opción tesis que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 8 de marzo de 2024

Atentamente,

Ing. Rafael Enrique Pérez Ordoñez MSc.
CI: 0916275076

AGRADECIMIENTO

En primer lugar, dar gracias a Dios por permitirme haber culminado este proyecto, agradecer a mis padres por el esfuerzo y el apoyo que me han brindado a lo largo de esta carrera universitaria y nunca dejarme solo. Gracias a todas las personas de la Universidad Salesiana por su atención y amabilidad en todo referente a mi vida como estudiante, a mis compañeros de clase durante todos los niveles de clase ya que gracias al compañerismo, amistad y apoyo moral han aportado a mis ganas de seguir adelante.

CARLOS AURELIO ORDÓÑEZ URGILES

AGRADECIMIENTO

En primer lugar, les agradezco todas las personas que de alguna manera directa o indirectamente han influido en mi vida. A mis padres por el apoyo incondicional, mis amigos y principalmente a los profesores que ayudaron a seguir con este logro personal.

JERRY BOLÍVAR RIVERA MURILLO

RESUMEN DEL PROYECTO

Año	Alumnos	Tutor de Proyecto de titulación	Proyecto de titulación
2024	Carlos Aurelio Ordóñez Urgiles y Jerry Bolívar Rivera Murillo	Ing. Rafael Pérez MSc.	Prototipo de incubación artificial para huevos de codorniz utilizando control difuso y tecnología IOT

En este proyecto de investigación se tiene como objetivo principal el diseño e implementación de un prototipo de incubación artificial utilizando control difuso y tecnologías IoT aplicados en huevos de codorniz, con la implementación de este tipo de proyectos se da paso a investigaciones futuras y desarrollos de nuevas tecnologías aplicadas a la avicultura específicamente a las codornices. Asi como también es de mucha utilidad para los conocimientos adquiridos de los futuros ingenieros electrónicos de la Universidad Politécnica Salesiana sede Guayaquil aplicados a desarrollos de tecnologías IoT.

En este proyecto se desarrolló un prototipo aplicado a la incubación de huevos de codornices utilizando sistemas de impresión 3D para las partes mecánicas y físicas de la incubadora de huevos, también se utiliza sensores de humedad relativa, temperatura, luminosidad y rotación de incubadora, con la monitorización de estos parámetros de manera remota utilizando servidores web en la nube que permiten la toma adecuada de decisiones para la incubación de huevos de codorniz con la finalidad del aumento de producción de este tipo de proteína.

En el desarrollo de este prototipo se utilizó un microcontrolador ESP32 para el procesamiento de datos que son obtenidos por los sensores, estos datos son administrados por el ESP32 y controlados para el envío a un servidor web en la nube como Ubidots. Posterior a la toma de datos se realiza un evaluación y análisis que sirve para generar alertas cuando los umbrales de temperatura, humedad o luminosidad varíen y afecten a la incubación de los huevos, para este caso se utiliza lógica difusa para el control de los parámetros ambientales dentro de la incubadora. El diseño del control de la temperatura y humedad relativa de la incubadora de huevos de codorniz se la realiza mediante código de programación quemado en el ESP32 y utilizando librería de Arduino PID_Fuzzy.

ABSTRACT

Year	Students	Degree Project Tutor	Technical Degree Project
2024	Carlos Aurelio Ordóñez Urgiles y Jerry Bolívar Rivera Murillo	Ing. Rafael Pérez MSc.	Prototype of artificial incubation for quail eggs using fuzzy control and IOT technology

The main objective of this research project is the design and implementation of an artificial incubation prototype using fuzzy control and IoT technologies applied to quail eggs, with the implementation of this type of project gives way to future research and development of innovative technologies applied to poultry farming, specifically to quail. It is also extremely useful for the knowledge acquired by future electronic engineers of the Salesian Polytechnic University in Guayaquil applied to the development of IoT technologies.

In this project, a prototype was developed applied to the incubation of quail eggs using 3D printing systems for the mechanical and physical parts of the egg incubator, relative humidity, temperature, luminosity and incubator rotation sensors are also used, with the monitoring of these parameters remotely using web servers in the cloud that allow adequate decision-making for the incubation of quail eggs with the purpose of increasing the production of this type of protein.

In the development of this prototype, an ESP32 microcontroller was used for the processing of data that is obtained by the sensors, this data is managed by the ESP32 and controlled for sending to a web server in the cloud such as Ubidots. After data collection, an evaluation and analysis are conducted that serves to generate alerts when the thresholds of temperature, humidity or luminosity vary and affect the incubation of the eggs, in this case fuzzy logic is used to control the environmental parameters inside the incubator. The design of the temperature and relative humidity control of the quail egg incubator is done by programming code burned in the ESP32 and using the Arduino PID_Fuzzy library.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLA

INTRODUCCIÓN

Los huevos de codorniz, gracias a su aporte proteico, pueden ayudar a prevenir alteraciones relacionadas con la masa magra. Sus proteínas de alto valor biológico, con todos los aminoácidos esenciales, tienen buena puntuación en cuanto a digestibilidad. Es importante asegurar un consumo de al menos 0,8 gramos de proteínas por kilo al día en personas sedentarias. Estos huevos pueden ayudar a prevenir la anemia debido a la presencia de hierro y de vitamina B12 en su interior (elsitioavicola.com, 2024).

En este proyecto de investigación se realizó el diseño e implementación de Prototipo de incubación artificial utilizando control difuso y tecnologías IoT aplicados en huevos de codorniz. En el proceso de investigación se evaluó la aplicación de algunos métodos de control, específicamente para el control de la temperatura y la humedad en la incubadora de huevos de codorniz.

Este prototipo consta de una estructura con material impreso en 3D el cual sirve para ubicar los huevos de codorniz y realizar la rotación de estos. Mediante control difuso se realiza la gestión de datos de temperatura ambiental, humedad relativa, luminosidad y rotación de los huevos. Los datos se envían a servidor web en la nube como Ubidots, y mediante esta se realiza el control a través de alertas cuando los parámetros superen los niveles de umbrales predefinidos. Otra ventaja de usar herramientas web en la nube es el almacenamiento de la información para futuro análisis y toma de decisiones.

Es importante considerar que este tipo de proyectos aplicados a la lógica difusa y a las tecnologías IoT permiten a los estudiantes de la carrera de ingeniería electrónica de la Universidad Politécnica Salesiana sede Guayaquil, profundizar los conocimientos adquiridos durante su preparación académica, específicamente en materias donde se abarquen temas de IoT.

1 El problema

1.1 Descripción del problema

El poco control que se tiene en la incubación de huevos de codorniz para la crianza de codornices en sectores rurales de la costa ecuatoriana donde la tecnificación avícola es escasa. Así como también las pocas herramientas que se tiene para el estudio de este campo investigativo relacionado a IoT y sistema de control aplicados a la crianza avícola de codornices.

1.2 Antecedentes

De acuerdo con la necesidad de la implementación de prototipo de incubadora IoT aplicados a huevos de codornices, se planteó un tema acorde a solventar los problemas de la poca tecnificación en la crianza de aves de codorniz, mediante incubadoras de huevos utilizando IoT y hardware de bajo costo.

1.3 Importancia y alcance

El trabajo de titulación propuesto es importante debido a que se buscó solucionar la baja tecnificación en la incubación de huevos de codorniz en zonas rurales de la costa del ecuador.

La realización de este proyecto tiene una aplicabilidad a corto plazo para los pequeños avicultores que se dedican a la crianza, producción y comercialización de aves de codorniz y huevos de codorniz para el consumo local en la zona costa del Ecuador.

1.4 Delimitación

1.4.1 Delimitación temporal

El tiempo estimado para el diseño, implementación y pruebas de funcionamiento del prototipo de incubadora de huevos fue de 3 meses, desde noviembre del 2023 a febrero del 2024.

1.4.2 Delimitación espacial

Las validaciones del funcionamiento y las demostraciones del prototipo IoT se realizó en el domicilio de los autores de esta tesis. El proyecto físico se entrega como prototipo de pruebas para la carrera de ingeniería electrónica de la Universidad Politécnica Salesiana sede Guayaquil campus centenario.

1.4.3 Delimitación académica

El diseño e implementación de prototipo de incubación artificial utilizando control difuso y tecnologías IoT aplicados en huevos de codorniz es muy importante para desarrollar las capacidades y destrezas de los futuros ingenieros electrónicos de la Universidad Politécnica Salesiana sede Guayaquil ya que sirve como guía práctica en laboratorio y para la investigación en este campo de la ciencia.

1.5 Beneficiarios de la propuesta

Este proyecto tiene como beneficiarios directos pequeños avicultores que busquen mejorar sus técnicas para la crianza de aves de codorniz.

Además, se ven beneficiados los estudiantes de la carrera de ingeniería electrónica de la Universidad Politécnica Salesiana sede Guayaquil, ya que se dispone de este prototipo para prácticas de laboratorio y futuros proyectos relacionados.

1.6 Propuesta de solución

Se propuso un prototipo para la incubación de huevos de codorniz utilizando sistema de control difuso para el control y monitorización de temperatura, humedad, rotación y luminosidad para la incubación de los huevos de codorniz.

Los datos obtenidos por los sensores son monitorizados en servidor web en la nube para control en tiempo real, como Ubidots.

Se configuró alertas que indiquen cambios en la temperatura y humedad de los huevos que

están en la incubadora mediante Ubidots.

Se propuso adicional una maqueta con impresión 3D, el modelado con fusión 360 y el laminado con el software PrusaSlicer para fabricar las bases que sostendrán a los huevos de codorniz, el cual se conectó a un rotor controlado por el ESP32.

1.7 Innovación e impacto del proyecto

El desarrollo de este proyecto de investigación tiene un nivel de innovación muy alto, debido a que existe la necesidad de la tecnificación de la crianza de codornices utilizando herramientas IoT y de control para la incubación de huevos de este tipo de aves. Este tipo de proyectos son de alto impacto tecnológico ya que une el IoT, el control automatizado, servidores web en la nube, y la impresión 3D de partes y piezas del prototipo.

1.8 Objetivos

1.8.1 Objetivo general

Elaborar un prototipo de incubación artificial utilizando control difuso y tecnologías IoT para ser aplicado en huevos de codorniz.

1.8.2 Objetivo específico

- Elaborar la estructura del prototipo de incubadora de huevos de codorniz mediante el uso de software para impresión en 3D.

- Diseñar el sistema de control difuso para el monitoreo y adecuación de parámetros de temperatura, humedad, luminosidad y rotación a ser aplicados en el prototipo de incubadora de huevos de codorniz.

- Implementar el sistema IoT para el monitoreo remoto de los diversos parámetros a considerar en el prototipo de incubadora de huevos de codorniz.

2 Fundamentos teóricos

2.1 Crianza de codorniz

El aumento del consumo de huevos de codorniz hace que el avicultor se preocupe por mejorar este segmento. En la actualidad hay un aumento sustancial en el número de granjas automatizadas, lo cual demuestra que la coturnicultura acompaña la evolución de las aves de corral en su conjunto (Elsitioavicola.com, 2023).

La coturnicultura, o cría de codornices, ha experimentado un notable proceso de mejora en sus instalaciones y en la genética de las aves. Estos avances son fundamentales para garantizar una producción eficiente y evitar pérdidas significativas, especialmente con el aumento en el tamaño de los lotes y de las parvadas de reproductoras.

La cría de codornices se ha convertido en una actividad económica atractiva por diversas razones. Entre ellas, su capacidad de adaptación a diferentes condiciones regionales, su bajo consumo de alimento y su alta producción de huevos son aspectos destacados. Además, los huevos de codorniz han ganado popularidad en el mercado alimenticio debido a su valor nutritivo y a la creciente demanda por parte de consumidores conscientes de la importancia de una dieta saludable.

Los empresarios con visión de futuro en el agronegocio han reconocido el potencial de este mercado y han invertido en la producción de huevos de codorniz, lo que ha contribuido aún más a su expansión. La codorniz es una especie productiva, fácil de manejar y con un rápido crecimiento, lo que la hace ideal para la cría en granjas. Además, sus huevos son ricos en proteínas, vitaminas y minerales, lo que los hace atractivos para aquellos que buscan opciones nutritivas en su dieta.

En resumen, la coturnicultura ha experimentado avances significativos tanto en las instalaciones como en la genética de las aves, lo que ha contribuido al crecimiento de esta actividad como una opción rentable en el agronegocio.

Determinándose que las mejores respuestas en cuanto a la calidad y peso del huevo (12,53 gr) y del cascaron (0,2636 mm) por lo tanto, tienen mayor ventaja para la incubación. En

cuanto a los factores de: manejo debe realizarse con el mayor cuidado posible para evitar rupturas o pérdidas de los huevos, para la luz se debe crear un plan de iluminación especial que permita elevar los rendimientos productivos de las aves, la humedad debe estar entre el 50 y 60 % para mitigar la muerte embrionaria. Por lo que se recomienda que los huevos de codorniz se deben incubar a una temperatura de 37.5°c y humedad de 45% en los primeros 15 días, aumentando hasta un 65% durante aproximadamente 3 días. En la figura 1 se observa la crianza de codorniz en jaula (Elsitioavicola.com, 2023).

Figura 1 Crianza de codorniz en jaula
(Elsitioavicola.com, 2023)

2.1.1 Factores que influyen en una buena cría

Pesaje – Es un proceso muy importante que se debe iniciar desde el primer día cuando llegan las pollitas. El proceso de pesaje semanal es una guía para los cambios en la alimentación y da una idea de la uniformidad y de si es necesaria una mayor selección.

Temperatura – Las aves necesitan una temperatura de 37°C a 38°C durante los primeros días, luego baje gradualmente hasta alcanzar la temperatura ambiente. Se requieren

calentadores automáticos para mantener la temperatura correcta para cada grupo de edad. El uso de calentadores de manos en ambientes muy reducidos requiere una supervisión constante para evitar el sobrecalentamiento.

Agua – El agua es un ingrediente importante porque las aves consumen el doble de agua que la que comen. Los cambios de agua siempre deben realizarse en el bebedero para evitar calentar el bebedero. El agua debe provenir de una fuente conocida y debe revisarse periódicamente para comprobar su calidad.

Alimento – La primera comida para codornices debe ser de grano fino para que se consuma de forma homogénea y equilibrada, evitando ingredientes selectivos y residuos como pellets de maíz en un comedero automático.

2.1.2 Sistemas de producción

Cría en piso y postura en jaula – La densidad total es de unas ochenta aves por metro cuadrado; cuanto más cerca, menor será la homogeneidad y menor la producción.

Cría en piso con recría y postura en jaula – La cría y crianza de codornices en terreno llano en la primera fase y en jaulas en la segunda fase proporcionará más consistencia.

Cría, recría y postura en jaula – Todavía es un proceso nuevo. Todavía es difícil conseguir jaulas que sean efectivas de principio a fin del proceso de la crianza.

Se usan partes de las jaulas para la cría el primer día debido a las mayores temperaturas requeridas, y luego reducir la densidad de las jaulas restantes. Este proceso de enjaulamiento tiene un aspecto positivo: las aves pueden seleccionarse tempranamente y así ayudar a su recuperación.

Cría en piso con sistema automático – Dada la capacidad de mantener un ambiente sin grandes cambios de temperatura y humedad, es un proceso que puede dar buenos resultados. El sistema es el mismo que para los pollos de engorde, pero se ajusta el número de equipos.

Galpones – Un diseño relativamente cerrado, con un sistema de cortinas o ventanas para ventilación, ayuda a proteger contra el frío en las primeras etapas del envejecimiento. Los ventiladores y los aspersores son útiles en los días calurosos y secos.

Las aves en este sistema se pueden criar en círculos o en habitaciones pequeñas, utilizando astillas de madera, un material absorbente, como lecho. Los equipos utilizados serán calefactores y podrán ser de gas, leña o eléctricos, y los comederos y bebederos serán de dos etapas, adaptándose a las distintas etapas de la cría.

2.1.3 Producción en jaula

Un punto positivo es la estandarización de las aves y el aumento de peso al inicio de la puesta, ya que habrá menos competencia entre aves. Una buena reproducción y crianza permitirá que el ave alcance su máximo potencial genético.

El manejo de codornices incluye las condiciones ambientales, equipos y procedimientos utilizados por el criador.

Se debe tener en cuenta la temperatura, la humedad, la luz, la densidad y los índices medios de los equipos. Cuando las instalaciones son inadecuadas y se utilizan mal, tienden a generar grandes cantidades de desperdicio de alimentos ya que las codornices tienden a limpiar con frecuencia.

Una forma de evitar el desperdicio es alimentar a los animales con más frecuencia y ajustar el equipo. Recortar el pico puede ayudar a reducir el desperdicio. Cuando se utilizan alimentos envasados, existe una alta probabilidad de generar residuos debido a que los roedores dañan las bolsas.

La presencia de aves silvestres puede resultar perjudicial, por lo que el granero debe estar bien protegido para impedir su acceso. Los roedores no sólo son una fuente de transmisión de enfermedades sino también una gran fuente de consumo de alimentos.

Por lo tanto, es importante realizar un seguimiento continuo para garantizar niveles aceptables.

2.1.4 Manejo

La necesidad de mantener altas temperaturas cuando se crían pollos por primera vez requiere que el gallinero sea lo más hermético posible, lo que dificulta el intercambio de aire. Tenga en cuenta que los calentadores de gas quemarán oxígeno, creando un ambiente inadecuado para las aves, por lo que es necesario controlar las persianas.

La humedad en el comienzo de la reproducción no es deseable porque produce amoníaco, que es perjudicial para las aves. Retire siempre la humedad y agite la cama.

El consumo normal de agua es aproximadamente el doble de la cantidad de agua ingerida y varía con la temperatura ambiental. La falta de agua provocará una decoloración grave de los huevos. Otros factores estresantes importantes incluyen la falta de comida y luz, ruidos inusuales, mala ventilación e incluso cambios en la vestimenta y la vestimenta de los trabajadores.

Inicialmente debe haber luz las 24 horas para facilitar la alimentación y evitar que las aves mueran por hacinamiento. En la etapa de crianza esto no es necesario porque el proceso de maduración ocurre más rápido, lo que dificulta alcanzar el peso de producción deseado. En la edad adulta, la luz estimula y acelera las funciones puberales y reproductivas.

La iluminación comenzó cuando las aves alcanzaron el 5% de producción, teniendo una duración inicial de 15 horas y aumentando gradualmente semanalmente hasta alcanzar las 17 horas. El día comienza a las 5 a.m., el día termina a las 10 p.m. Para evitar sombras es importante una buena distribución de la iluminación. La figura 2 muestra una codorniz con huevos (elproductor.com, 2024).

Figura 2 Codorniz y sus huevos
(elproductor.com, 2024)

2.2 Sistemas de control difuso para control de parámetros ambientales

La conocida como Fuzzy Logic es la lógica que utiliza expresiones que no son ni completamente ciertas ni falsas.

La Fuzzy Logic se aplica a conceptos que pueden adquirir un valor cualquiera de veracidad dentro de un conjunto de valores que oscilan entre dos extremos: La verdad absoluta, La falsedad total (*Lógica Difusa o Fuzzy Logic: Qué Es y Cómo Funciona + Ejemplos*, n.d.).

Es importante recordar que no es la lógica difusa en sí misma sino el objeto que se estudia. Esto se debe a la falta de definición del concepto al que se refiere. La lógica difusa le permite manejar información imprecisa. Por ejemplo, «temperatura baja» o «estatura media», en términos de conjuntos que no son claros. Esta cuantificación de los adjetivos será la base que se va a intentar tratar mediante la Fuzzy Logic.

Las aplicaciones de la Fuzzy Logic se pueden ampliarlas a los sistemas expertos o desarrollar controladores difusos. Algún ejemplo de controlador difuso sería el controlador de un termostato. Lo primero se debe de tener en cuenta para definir un controlador difuso es que hay que definir las particiones. Generalmente, estas particiones tienen que ser completas. Es decir, tienen que cubrir todo el rango de valores posible para el problema que se pretende resolver.

Por otro lado, también es habitual definir un solapamiento entre variables de entre un 20% o un 30%. En el ejemplo de las temperaturas, este solapamiento se encontraría, por ejemplo, en los 24 grados. Los 24 grados pueden ser una temperatura templada pero también podrían acercarse a ser una temperatura cálida. También es importante definir las reglas difusas, que permiten unir los conjuntos borrosos de entrada con los conjuntos difusos de salida. En la figura 3 se observa un ejemplo de lógica difusa, utilizando sensor de temperatura (*Lógica Difusa o Fuzzy Logic: Qué Es y Cómo Funciona + Ejemplos*, n.d.).

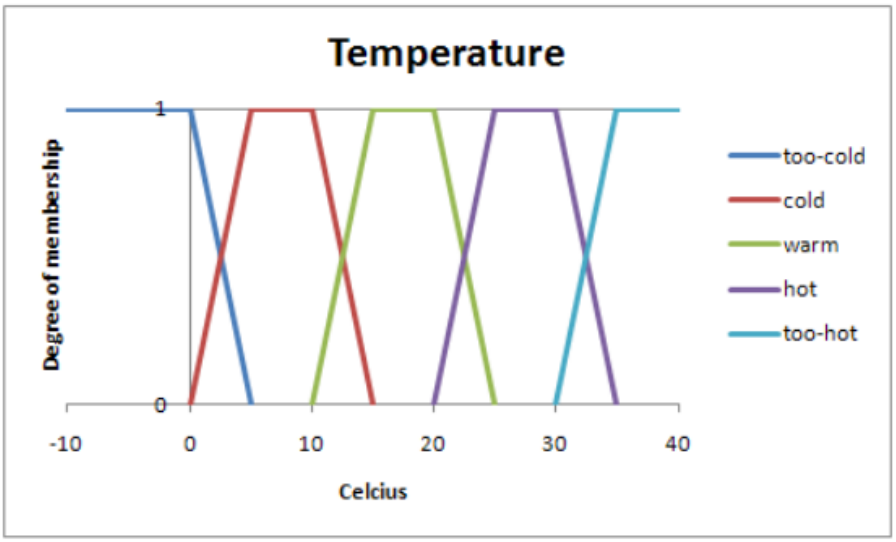


Figura 3 Ejemplo de lógica difusa

(Bello, 2021)

2.3 Controlador PID

La lógica difusa surge de la necesidad de formalizar en el lenguaje matemático algunas situaciones cotidianas que son imprecisas o no tienen valores claros que no puedan cuantificarse numéricamente. Por ejemplo, la mente humana está construida con elementos lingüísticos más que con números; si la lógica Es una ciencia que intenta presentar razonamientos y debe tener en cuenta estos factores.. La lógica difusa, en particular, es muy conveniente para representar conocimientos y datos tan imprecisos. En la lógica clásica un elemento pertenece o no a un determinado conjunto, pero si como ejemplo se asume que la altura de una persona se clasifica en un conjunto determinado, entonces sería injusto afirmar que una persona con una altura de 1,79 m pertenece al grupo de "altura media", el otro 1,80 m pertenece al grupo de "altura alta". En lógica difusa esta situación se soluciona porque allí se tiene en cuenta la pertenencia al conjunto, es decir, un elemento puede pertenecer en cierta medida a más de un conjunto.

Aplicando la tecnología, es necesario desarrollar sistemas que tomen decisiones basadas en un conjunto de variables de entrada. La lógica difusa ofrece soluciones interesantes a este tipo de situaciones., especialmente en aquellas donde los valores de las variables que determinan la salida se pueden agrupar en conjuntos. Un ejemplo de tal aplicación es el control de variables físicas. El control de variables físicas como temperatura, humedad, luminosidad, etc.

En todos los sistemas de control se establece un valor predeterminado de la variable física de interés, este valor se llama setpoint en inglés. El valor de una variable se mide en un sistema físico y se considera la diferencia entre el valor medido y el valor establecido. Esta diferencia se llama error sistemático etc., y la forma en que se maneja determina el tipo de control. Entre los sistemas de control clásicos, el más común es el Derivado Integral Proporcional (PID), como se muestra en la Figura 4. Donde la parte proporcional (P) reacciona en función del error actual del sistema; el valor integral (I) genera su salida en función de la suma de los cambios de error en el tiempo, y la acción derivativa (D) determina la acción en función de la tasa de cambio temporal del error. Los sistemas PID se utilizan ampliamente en sistemas de control como temperatura, humedad y velocidad del motor. (García-Sucerquia & Palacio-Gómez, 2011).

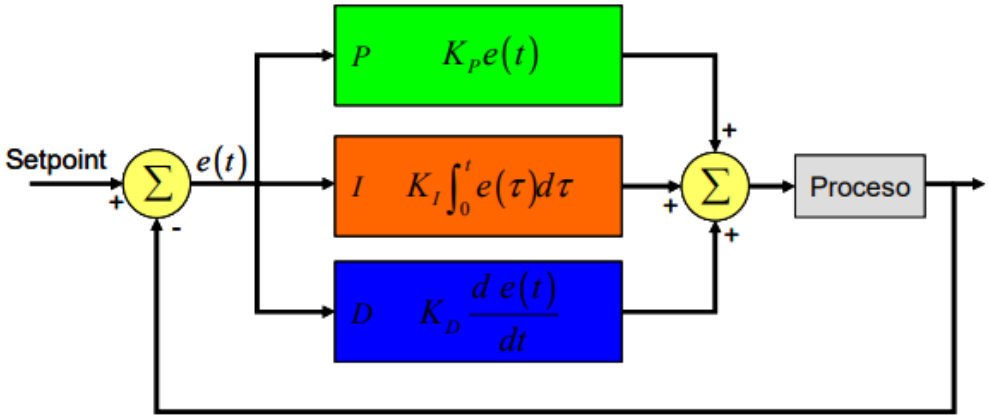


Figura 4 Controlador PID

(García-Sucerquia & Palacio-Gómez, 2011)

El proceso de optimización del sistema de control PID se lleva a cabo calculando las constantes del sistema Kp, KI y KD, ver Figura 4. El proceso se puede realizar en una primera aproximación modelando el sistema a controlar, lo que suele tener un coste computacional elevado.

La finalización se realiza mediante un proceso de prueba y error, lo que en algunos casos constituye una limitación del control PID. Una forma de control más moderna es el control de lógica difusa. La lógica difusa es una forma de lógica polinómica que se ocupa de un razonamiento aproximado en lugar de preciso. Al contrario de la lógica numérica, donde las variables sólo tienen el valor 1 o 0, las variables en la lógica difusa pueden tomar cualquier valor entre estos dos valores extremos. Este rango ampliado de valores permite que las variables difusas tengan grados de pertenencia, porcentajes de verdad o permitan que las afirmaciones se califiquen en un rango más amplio que el de verdadero o falso. Por estas características, la lógica difusa se adapta mejor al mundo en el que se vive, ya que permite interpretar expresiones de la vida cotidiana como "está muy claro", (García-Sucerquia & Palacio-Gómez, 2011).

2.4 Controlador difuso

El algoritmo de control se basa en un controlador difuso y se muestra en la Figura 5. El controlador difuso consta de cuatro elementos (Liu & Zhang, 2003).

- **Regla base:** Contiene la cuantificación de lógica difusa basada en la experticia sobre cómo obtener control bueno.

- **Mecanismo de inferencia:** Imita las decisiones de los expertos para interpretar y aplicar mejor el conocimiento para controlar un proceso.

- **Interfaz de fusificación:** Convierte la entrada del controlador en información que el motor de inferencia pueda utilizar para ejecutar y aplicar reglas.

- **Interfaz de desfusificación:** Convierte todas las conclusiones de cada mecanismo de inferencia en entradas para el proceso.

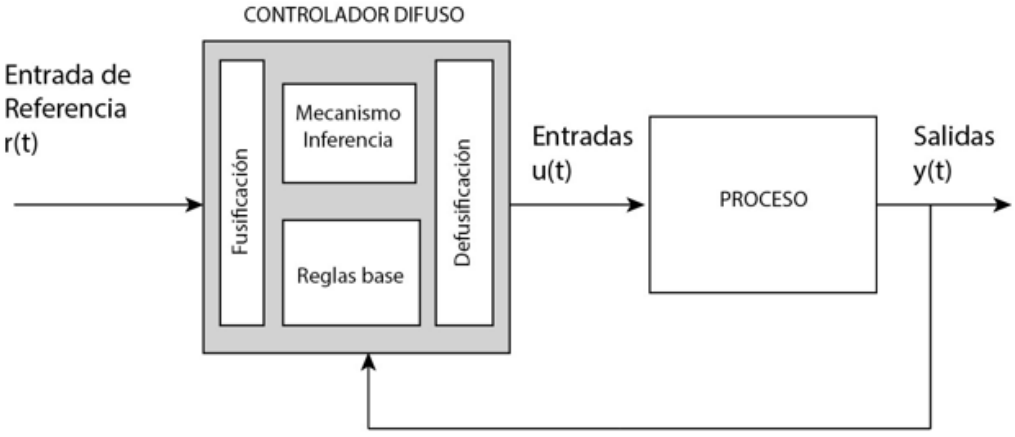


Figura 5 Controlador Difuso
(Liu & Zhang, 2003)

Las ventajas de los controladores difusos son obvias para problemas muy complejos en los

que existe una idea intuitiva de cómo lograr un alto rendimiento del controlador. En estas aplicaciones, la obtención de modelos matemáticos precisos es muy compleja debido a la gran cantidad de modelos matemáticos necesarios. entradas y salidas, o debido a sus procesos no lineales o estocásticos (Liu & Zhang, 2003).

El módulo de fusificación realiza las siguientes funciones:

- Convierte las variables físicas de la señal de proceso y la señal de error en un subconjunto difuso normalizado, que consta de un intervalo del rango de valores de entrada y una función de membresía normalizada que describe la confianza de la entrada que pertenece a este rango.

- Selecciona una función de membresía óptima, buena, razonable e ideal según algunos criterios convenientes para la aplicación.

La estructura general de un controlador de lógica difusa consta de tres elementos principales: una unidad difusa en la entrada, un proceso de razonamiento basado en las reglas básicas del control difuso y una unidad de defusificación en la salida. (Chen & Tat Pham, 2024) como se muestra en la Figura 6.

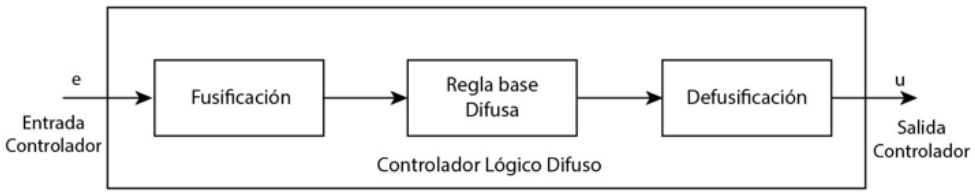


Figura 6 Estructura general de un controlador lógico difuso
(Chen & Tat Pham, 2024)

2.4.1 Módulo de fusificación

Se establecen las funciones de membresía en la figura 7 para la señal del error e, cuya ecuación se presenta a continuación.

$$e(t) = r - y(t)$$

Donde:

r: Punto de consigna
y(t): Salida de la planta

Se utiliza la siguiente notación en las funciones de membresía durante el diseño del control difuso.

- NS: Negativo pequeño
- NL: Negativo largo
- ZU: Cero

- PS: Positivo pequeño
- PL: Positivo largo

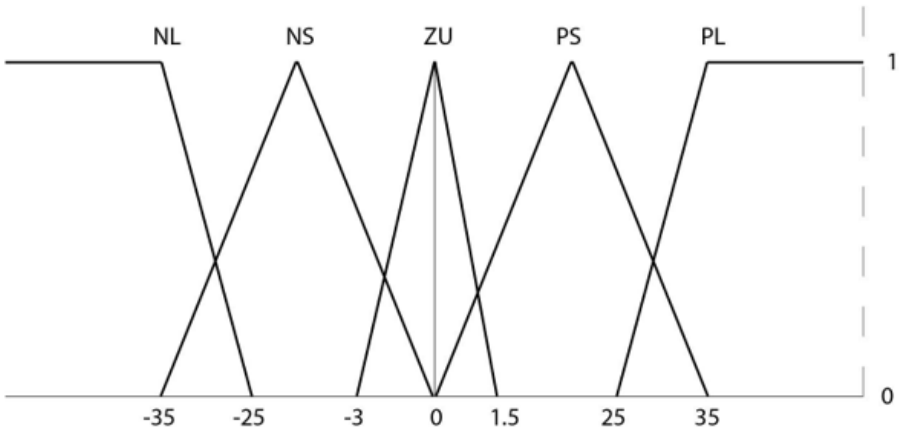


Figura 7 Funciones de membresía para el error

(Chen & Tat Pham, 2024)

2.4.2 Reglas base de la lógica difusa

Se establecen las entradas del controlador, primero se dispone del error e.

La señal de error es una variable de entrada del controlador, sin embargo, para tener una regla básica, es necesario que haya más variables de entrada auxiliares, por lo que se considera el cambio de la señal de error e, lo que ayuda a distinguir si la curva está subiendo o bajando. La ecuación para la derivada del error se muestra a continuación:

$$e\,(t) = r^{\cdot} - y(\dot{t})$$

Por lo tanto, se considera que la función de pertenencia es la derivada del error que se muestra en la figura 8. Estas funciones de membresía se eligen en función de la aplicación de temperatura en cuestión y buscando un valor a partir del cual se debe modificar la salida del controlador para evitar la inercia térmica debido a la resistencia de sobrepulso.

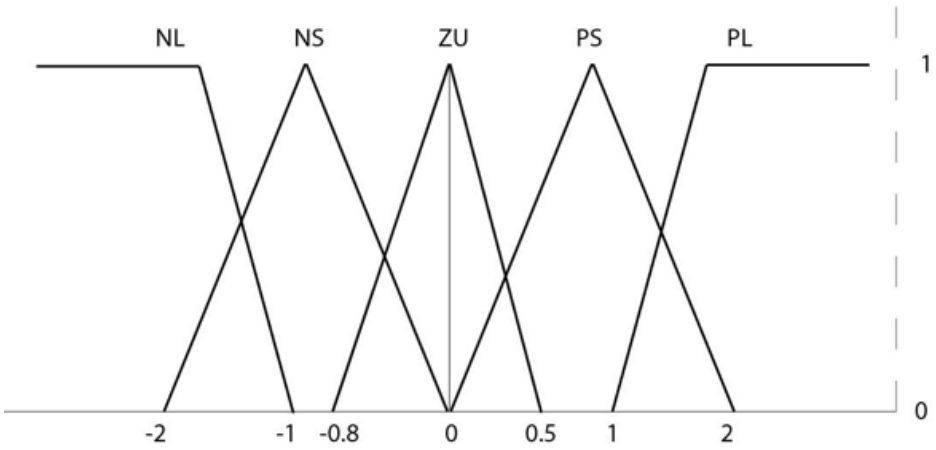


Figura 8 Funciones de membresía para la derivada error

(Liu & Zhang, 2003)

Con el fin de determinar la tasa de cambio máxima se efectúan pruebas, encendiendo el prototipo específicamente el actuador, se empieza con una temperatura inicial ambiente de 30°C que puede fluctuar hasta 37°C.

Posteriormente, con el fin de facilitar la implementación del algoritmo difuso en el código del ESP32, se separan las funciones de membresía en tres tipos de formas principales que se presentan en la figura 9.

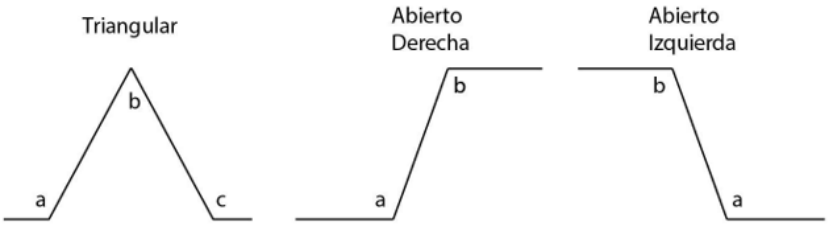


Figura 9 Formas bases de las funciones de membresía

La figura 10 muestra las formas abiertas a la derecha e izquierda con la notación adicional requerida para la obtención de las reglas base.

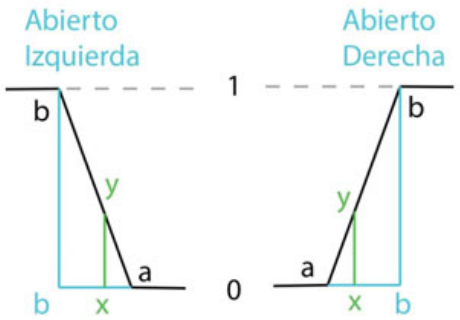


Figura 10 Formas abiertas a la derecha e izquierda para análisis del valor de membresía

(Liu & Zhang, 2003)

2.5 Sensores Ambientales IoT

Los sensores ambientales realizan mediciones de alta precisión de diferentes parámetros climáticos, como, temperatura, humedad, calidad el aire, entre otros. Son de uso esencial para la toma de acciones preventivas y correctivas en diferentes áreas de interés, las más comunes son áreas verdes o zonas escolares. El sensor de temperatura y humedad DHT22 detecta y mide temperatura (hasta 50°C) y humedad (hasta 90% RH).

DHT22 (AM2302) es un sensor de temperatura y humedad relativa con buen rendimiento y bajo costo. Contiene un detector de humedad capacitivo y un termistor para medir el aire ambiente y mostrar los datos a través de una señal en el pin..

Utilizar el sensor DHT22 en la plataforma Arduino/Raspberry Pi/Nodemcu es muy sencillo, tanto a nivel de software como de hardware. A nivel de software, Arduino proporciona una biblioteca que admite el protocolo "bus único". En cuanto al hardware, simplemente conecte el pin VCC de alimentación a 3-5 V, el pin GND a tierra (0 V) y el pin de datos a los pines digitales del Arduino. Si desea conectar varios sensores DHT22 a un Arduino, cada sensor debe tener su propio pin de datos. Quizás el único inconveniente del sensor es que sólo puede adquirir nuevos datos cada 2 segundos. Cada sensor se calibra en fábrica para obtener factores de calibración escritos en su memoria OTP, lo que garantiza una alta estabilidad y confiabilidad a largo plazo.

El DHT22 presenta mejores prestaciones respecto al sensor DHT11, como mejor resolución, mayor precisión y un empaque más robusto. En la figura 11 se observa el sensor de temperatura DHT22 (Naylampmechatronics.com, 2024).

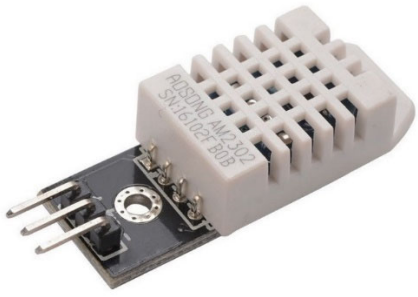


Figura 11 Sensor de Temperatura y Humedad DHT22
(Naylampmechatronics.com, 2024)

2.6 ESP32

ESP32 es una serie de SoC (por sus siglas en inglés, System on Chip) y módulos de bajo costo y bajo consumo de energía creado por Espressif Systems.

Los tipos de esp32 incluye los chips:

- ESP32-D0WDQ6
- ESP32-D0WD
- ESP32-D2WD
- ESP32-S0WD

En la figura 12 se observa el ESP32 (programarfacil.com, 2024)

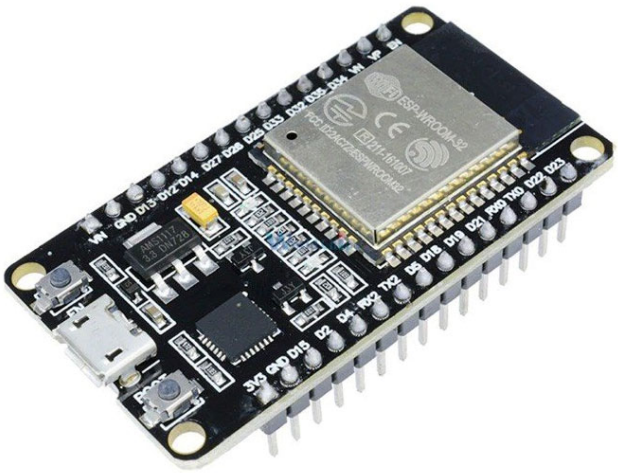


Figura 12 ESP32
(programarfacil.com, 2024)

2.7 Ubidots

Ubidots se define como una plataforma de IoT que empodera a innovadores e industrias, se enfoca a facilitar el desarrollo y escalamiento de prototipos para la producción, monitorización y desarrollo. Esta plataforma ayuda a enviar datos a la nube desde cualquier tipo de sensor conectado a internet. En la figura 13 se observa el dashboards de Ubidots (Bitcu.co, 2024).

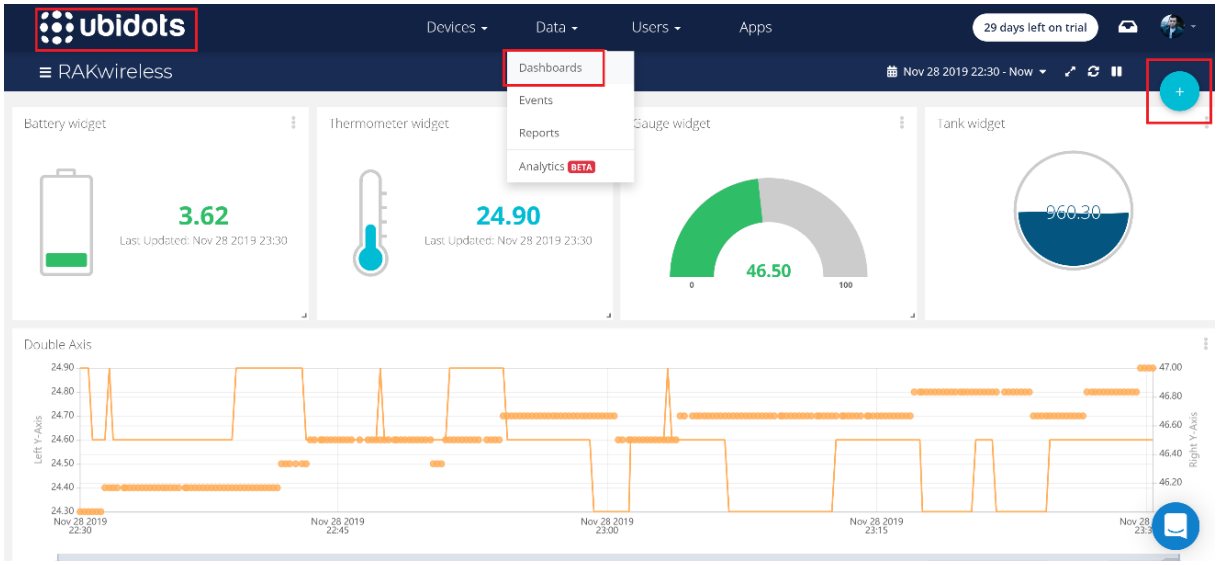


Figura 13 Ubidots
(Bitcu.co, 2024)

2.8 Impresión 3D

La impresión 3D es el proceso de creación de objetos mediante el depósito de capas de material unas sobre otra. Cuando se utiliza en la fabricación industrial, la impresión 3D se denomina fabricación aditiva (AM) a diferencia de los métodos sustractivos tradicionales como el fresado CNC.

Esta tecnología existe desde hace unos cuarenta años. Aunque la impresión 3D era inicialmente una tecnología lenta y costosa, los grandes avances tecnológicos han hecho que los métodos de fabricación aditiva actuales sean más asequibles y rápidos que nunca. Utilice un software especial para dividir el modelo 3D digital en cientos de capas delgadas y exportarlo en formato de código G.

Las capas se imprimen consecutivamente en 3D de una en una hasta obtener el objeto completamente impreso.

La creación de prototipos, una de las formas más populares de impresión 3D profesional, se puede realizar en el sitio prácticamente sin tiempo de entrega, y las iteraciones del diseño se pueden implementar e imprimir en el sitio. Esta tecnología también ofrece varias opciones para materiales de impresión 3D.. En la figura 14 se observa una impresión en 3D (3ds.com, 2024).

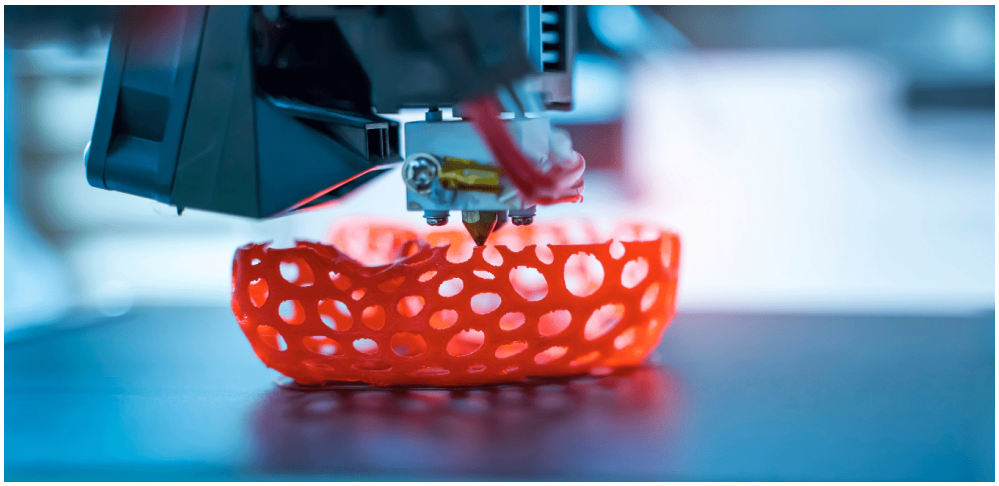


Figura 14 Impresión 3D
(bbva.ch, 2024)

3 Marco metodológico

3.1 Tipo de investigación

El proyecto se basa en una investigación exploratoria y experimental. Exploratorio porque el problema no ha sido estudiado anteriormente, sugiriendo que apenas está comenzando, con el objetivo de comprenderlo mejor y responder a las preguntas qué, por qué y cómo. Experimental, porque los datos se obtendrán a través de experimentos, en este caso particular será a través de sensores dentro de la incubadora de huevos de codorniz.

3.2 Diseño de investigación

El diseño de esta investigación es completamente para experimentar porque la pregunta establecerá una relación. En una incubadora de huevos de codorniz se controla, por ejemplo, de parámetros ambientales como la temperatura, la humedad relativa y la luminosidad. Este es un diseño de prototipo muy práctico porque ayuda a superar los problemas planteados en esta investigación.

3.3 Enfoque de la investigación

Esta metodología de investigación es mixta, es decir cualitativa y cuantitativa. Cualitativa, porque se recopila información de diversas fuentes como libros, artículos científicos, tesis y otras publicaciones, luego se realiza un análisis cualitativo de la información obtenida.

3.4 Metodología de investigación

A continuación, se explica la metodología del proyecto de titulación de acuerdo con los objetivos específicos.

Se realiza diseño de maqueta de incubadora de huevos de codorniz utilizando materiales de bajo costo y piezas con impresión 3D, las mismas que son conectadas al rotor el cual es comandado por el ESP32.

Se implementa un sistema IoT para la monitorización remota de la temperatura, humedad relativa, luminosidad y rotación de los huevos de codorniz utilizando servidor web en la nube como Ubidots, se crea una cuenta para acceso remoto mediante laptop o dispositivo móvil.

Se configura alertas de parámetros de temperatura, humedad relativa, luminosidad y rotación del prototipo IoT utilizando las reglas de Ubidots.

Se utiliza un sistema de control difuso para el control de temperatura, humedad y luminosidad de prototipo IoT de incubadora de huevos de codorniz, utilizando los sensores y actuadores para el control y manipulación de los parámetros a medir.

Para la maqueta se utiliza materiales de bajo costo e impresión 3D para los repositorios de los huevos de codorniz.

3.5 Proyectos de investigación vinculados

SISTEMA DE CONTROL DIFUSO PARA EL MONITOREO DE LA TEMPERATURA, LA HUMEDAD, EL PH Y LA CONDUCTIVIDAD ELÉCTRICA EN INVERNADEROS DE PLANTAS ORNAMENTALES

Autor: Eduardo Flores Gallegos
Tecnológico Nacional de México

SISTEMA DE CONTROL DE PARÁMETROS AMBIENTALES Y ALIMENTICIOS EN UNA CRIADORA DE POLLOS, BASADO EN LÓGICA DIFUSA

Autores: Segundo Lovera, José Félix
Universidad Autónoma del estado de México

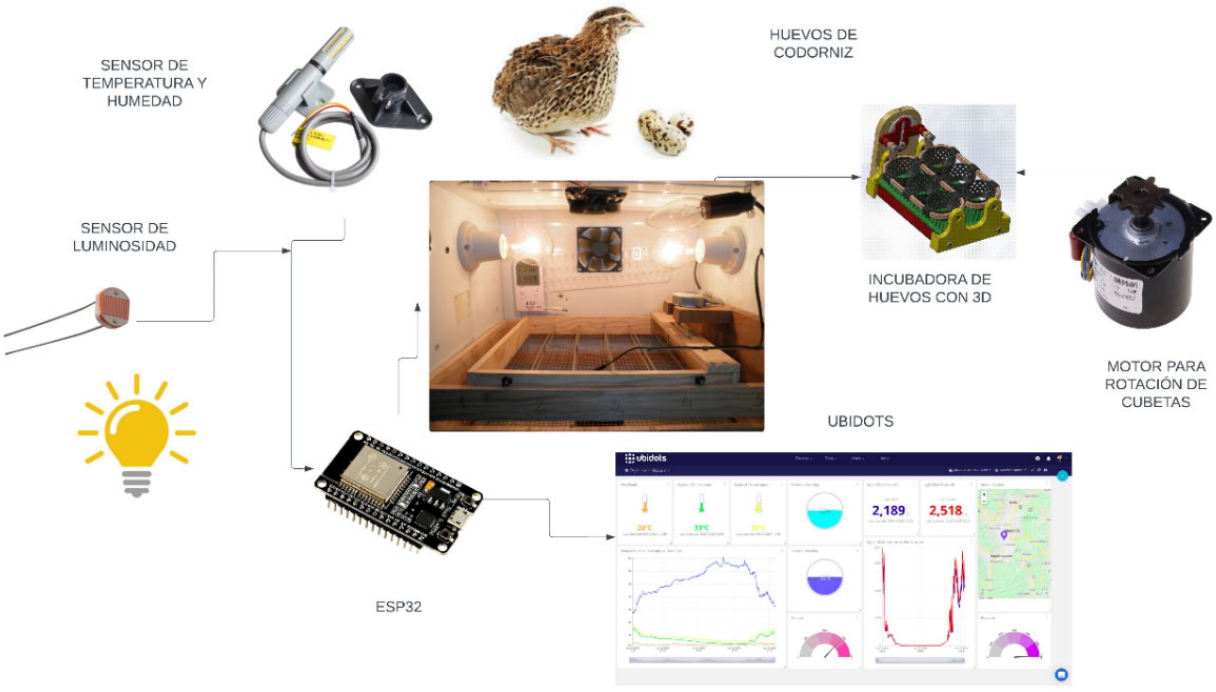
DISEÑO DE UN PROTOTIPO DE INCUBACIÓN ARTIFICIAL CON SISTEMA DE CONTROL DIFUSO PARA LA PRODUCCIÓN DE AVES DE CODORNIZ

Autora: César Cruz, Victor Vargas
Universidad de San Martin de Porres

3.6 Descripción de la propuesta

Los huevos de codorniz son un alimento saludable que se puede incluir periódicamente en la dieta. Son más pequeños que el pollo, tienen un sabor muy similar y son más densos desde el punto de vista nutricional porque contienen más calorías y tienen un alto contenido de ciertos minerales esenciales. Son muy prevenibles déficit. Por tal motivo es de suma importancia para los pequeños y medianos productores avícolas, contar con herramientas tecnológicas que mejoren la productividad de esta ave. A nivel local el pequeño productor avícola cuenta con poca tecnificación para el control en la incubación de huevos de codorniz, disminuyendo su producción y comercialización de estas aves en el mercado local.

En la figura 15 se muestra un esquemático del prototipo IoT de incubadora de huevos de codorniz, se indica los elementos que corresponden al prototipo diseñado para mejorar la población de aves de codorniz mediante la tecnificación en el cuidado de los huevos de codorniz.



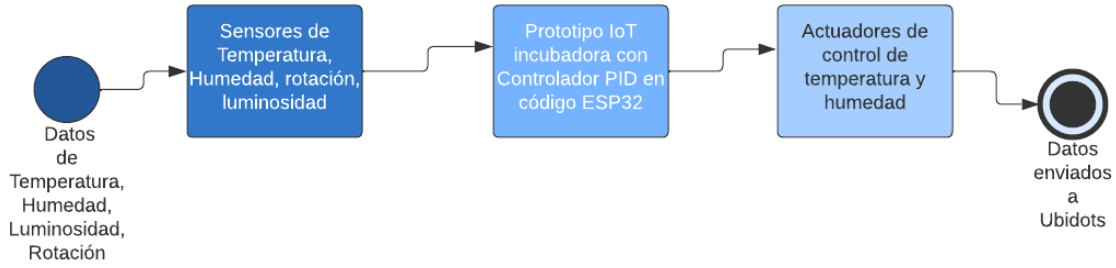


Figura 15 Esquemático de prototipo de incubadora de huevos de codorniz

3.7 Diseño de PID Fuzzy en ESP32

Los parámetros de control para tener en cuenta en la incubadora artificial son la temperatura, la humedad del ambiente y la rotación. Estos parámetros o variables físicas son controlados por un diseño de control y verificados por la simulación para satisfacer las características óptimas de cría de codorniz. Hacer una incubadora de huevos de codorniz que sea eficiente y económica es un desafío porque los embriones de huevos son muy frágiles, y cualquier cambio repentino en la temperatura o la humedad puede afectar significativamente el crecimiento de los pollitos y el tiempo para eclosionar.

Dentro de la industria generalmente se conocen tres tipos de controles de temperatura, ON-OFF, los cuales funcionan de acuerdo con su nombre, se enciende y apaga continuamente para mantener el punto exacto de ajuste de temperatura deseado. Cuando la temperatura excede el punto de ajuste, se apaga y cuando la temperatura está por debajo del punto de ajuste, se enciende. Los controladores proporcionales reducen la potencia del calentador cuando detecta que la temperatura excederá el punto de ajuste, manteniendo una temperatura estable y una respuesta rápida, pero puede exhibir oscilaciones en la variable controlada o presentar error en estado estacionario, y los controladores PID ofrecen control proporcional, integral y derivado, es control de retroalimentación para hacer que el error entre la salida y la entrada sea cero, compensa los cambios de temperatura gracias a su combinación y puede ajustar cada variable, haciéndola más precisa.

De los tipos de controladores, el más adecuado para el control de la temperatura de la incubadora es el controlador PID, que es el más susceptible a los cambios y puede reaccionar rápidamente para compensarlos. Además, puede predecir cualquier cambio o efecto en la salida, gracias a su función derivada. Actualmente, ya han realizado sistemas de control de temperatura para plantas de incubación, por ejemplo, se han utilizado controles basados en microcontroladores que pueden controlar la temperatura, la humedad y revertir los huevos de forma automática y a través del sistema de Internet de las Cosas (IoT) ayudar a los agricultores a monitorear la planta de incubación en tiempo real y de forma remota (Cruz & Vera, 2021).

También se han utilizado microcontroladores que han sido diseñados utilizando el control difuso, que consiste en determinar lógicamente qué proceso hacer para alcanzar los objetivos de control a partir de una base proporcionada por el diseñador. Esto controla la posición de los huevos, la temperatura y la humedad de la incubadora y garantiza las mejores condiciones para los diferentes tipos de huevos.

Para poder trabajar con la lógica Fuzzy se hizo uso de la librería PID_FUZZY_v1.h de la tarjeta ESP32; además se declararon las siguientes variables que son aquellas sobre las cuales se ejerce el control puesto que tienen la información para poder mantener la incubadora dentro de los parámetros deseados y son de tipo flotante:

- Temperature_1
- Temperature_2
- Humidity_1
- Humidity_2
- Temp_average
- Humidity_average

Los puntos de setpoint tanto para humedad como para temperatura fueron establecidos de acuerdo con los manuales de crianza del ave de codorniz; así como también el tiempo total que es de 18 días. Para ejercer el control fuzzy las variables que realimentan al mismo son: Input_t, Output_t, Setpoint_t, Kp_t, Ki_t, Kd_t, para temperatura y para humedad son: Input_h, Output_h, Setpoint_h, Kp_h, Ki_h, Kd_h

Es importante mencionar que los valores de setpoint dependerá del número de días que los

huevos lleven en la incubadora; es decir si es mayor o igual a 8 días se tendrá:

Setpoint_t=38.8
Setpoint_h=55
Caso contrario serán
Setpoint_t=38.3
Setpoint_h=55

Si es mayor a 15 días el setpoint de humedad es 65.
El valor promedio tanto de la humedad como de la temperatura permiten el encendido de los actuadores para el control de temperatura y humedad.

Se detalla el código de la librería PID_Fuzzy.h en la sección de Anexos. Esta librería cambia los valores el cual define las funciones de membresía del controlador PID.

La siguiente figura 16 detalla el funcionamiento del algoritmo basado en código del ESP32. El algoritmo inicia con la variable de control que en este caso es la lectura de los sensores de temperatura y humedad, se realiza el cálculo del error el cual ingresa al controlador, este valor a su vez ingresa en el algoritmo PID y fuzzy para posteriormente establecer un nuevo clico de trabajo.

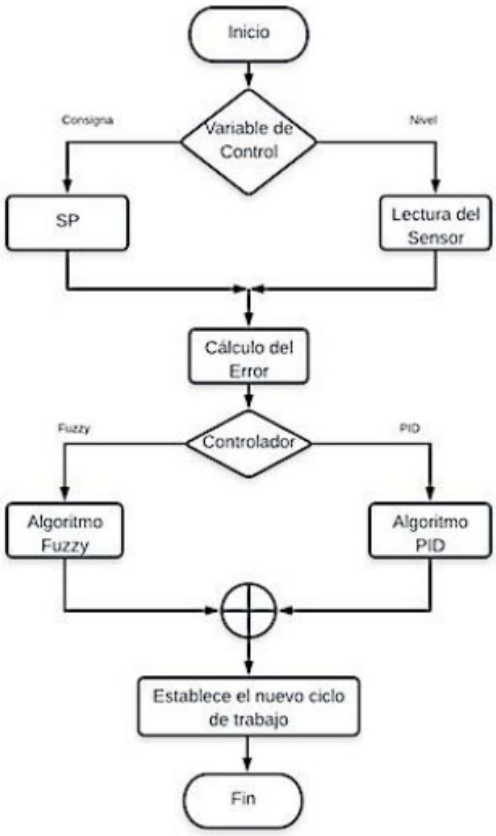


Figura 16 Diagrama de flujo del controlador PID

En las figuras 17 y 18 se observan los diagramas de bloques del control del sistema de temperatura y humedad relativa.

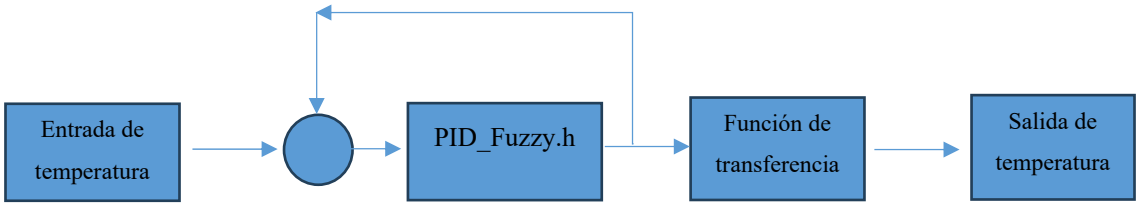


Figura 17 Diagrama de bloques del control del sistema de temperatura

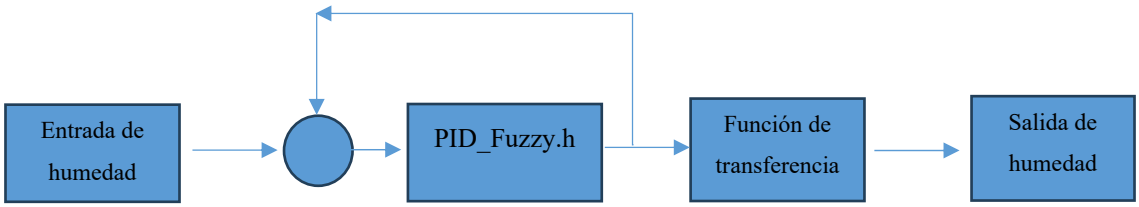


Figura 18 Diagrama de bloques del control del sistema de humedad

El cálculo del error se realiza afinando las salidas observables desde Ubidots. Estas gráficas se detallan en la sección de resultados.

3.8 Diseño electrónico de prototipo IoT

En esta sección se describe el diseño electrónico del prototipo IoT, tal como se muestra en la figura 19. Se observa el diseño del esquemático electrónico de la incubadora de codorniz con control difuso. Se observa las conexiones de la boya con nivel, el buzzer, reguladores, salidas mosfets, salidas servo relés, sensores, convertidora USB-UART y el ESP32.

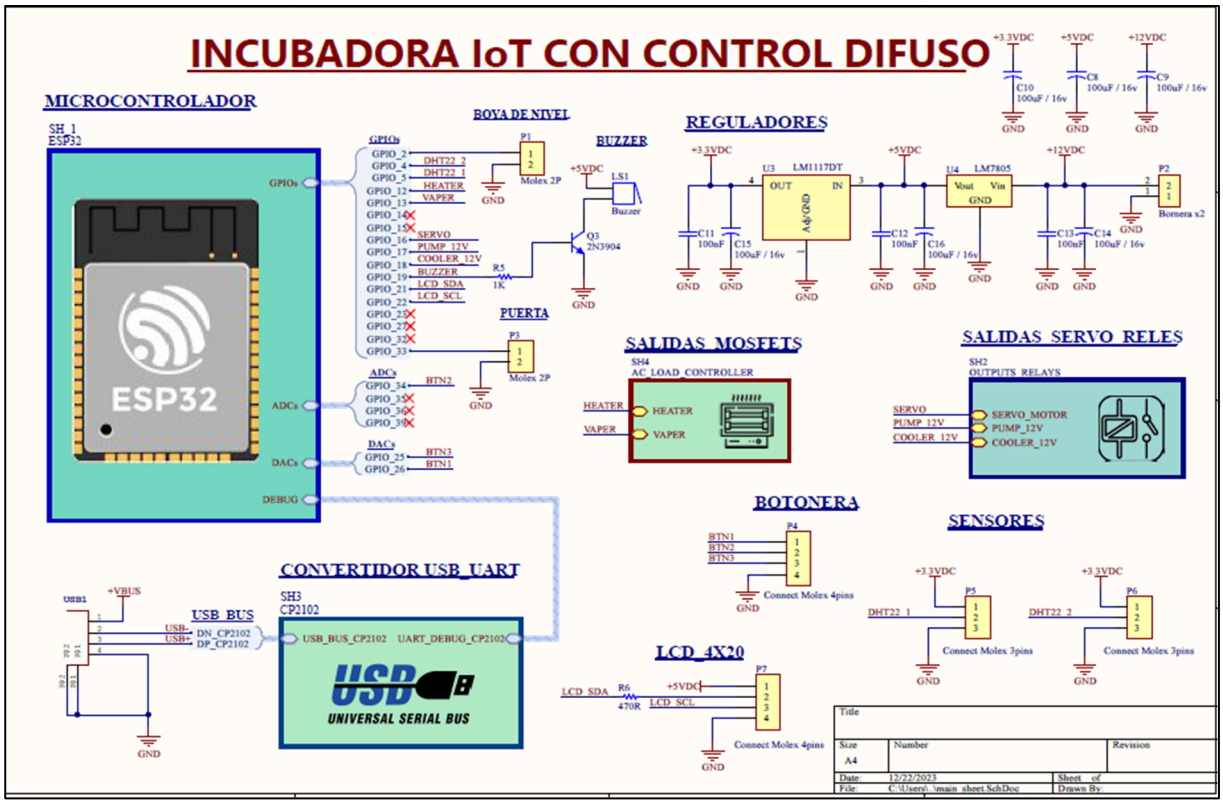


Figura 19 Esquemático electrónico de prototipo de incubadora

En la figura 20 se observa la distribución de pines del ESP32 WROOM-32U. Es importante identificar cada pin de salida y de entrada del ESP32, durante el diseño se predefinen que pines GPIO se utiliza, así como también los pines ADC, DAC y Debug se identifican plenamente previo al diseño de la PCB.

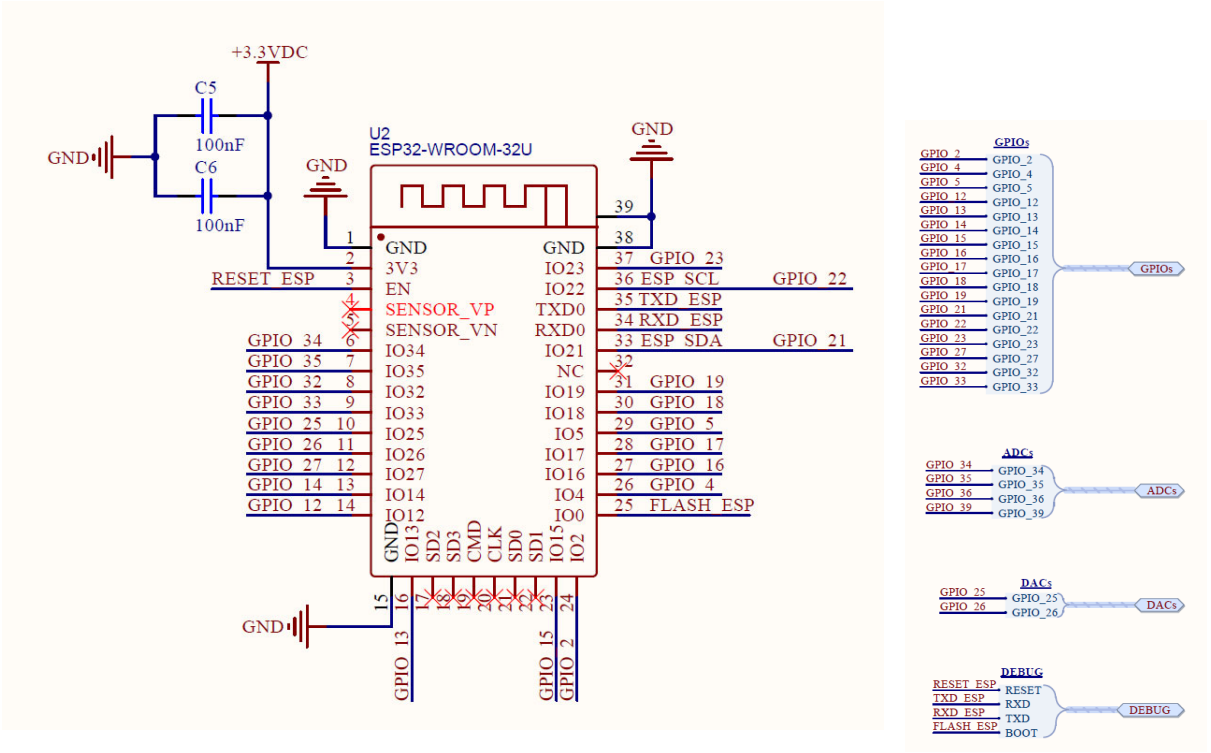


Figura 20 Distribución de pines ESP32

En la figura 21 se observa las conexiones del módulo USB-BUS. Se utiliza los pines 4 y 5 como puertos de comunicación BUS, se aprecia también el uso de condensadores en los pines asi como tambien las entradas de voltaje de 3.3 VDC.

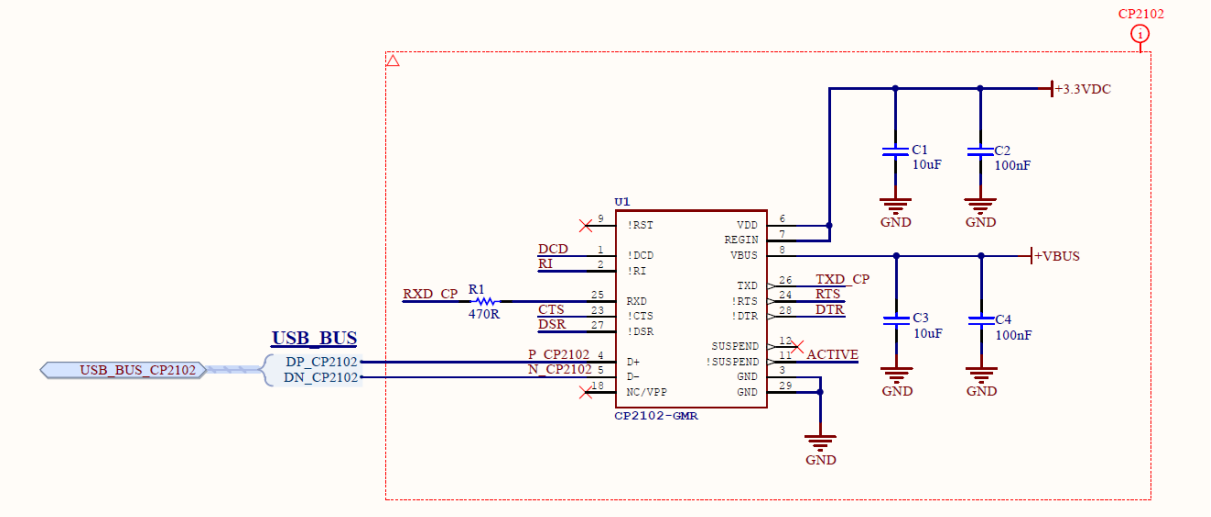


Figura 21 Conexiones USB-BUS

En la figura 22 se observa las conexiones de pines del módulo UART Debug CP2102. Estos pines se definen como TXD, RXD, Reset y Boot.

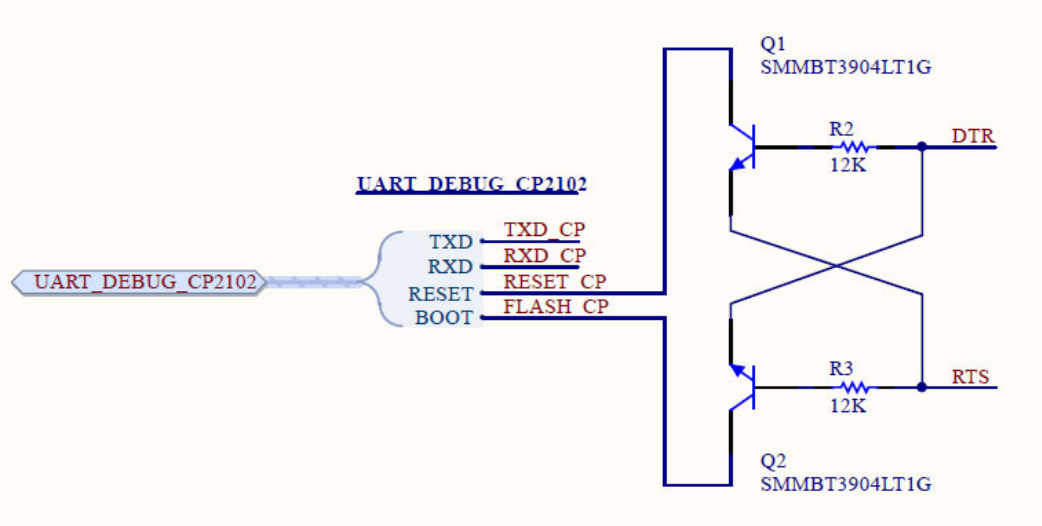


Figura 22 Conexiones UART

En la figura 23 se observa las conexiones de pines del servomotor. Se utiliza un controlador LM350T a 12 VDC el cual activará el servomotor.

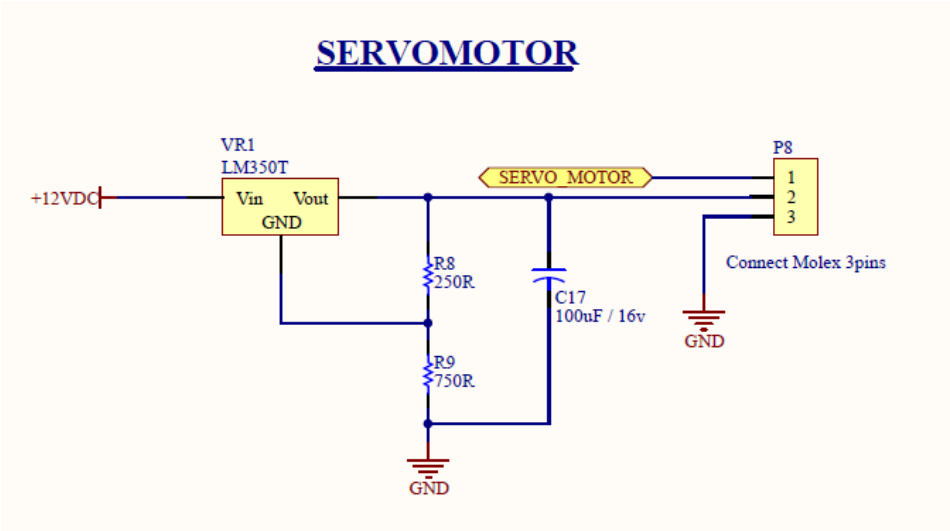


Figura 23 Conexiones del servomotor

En la figura 24 se observa las conexiones de pines de la bomba de agua de 12 VDC. Se observa tambien que se utiliza un relay de bobina.

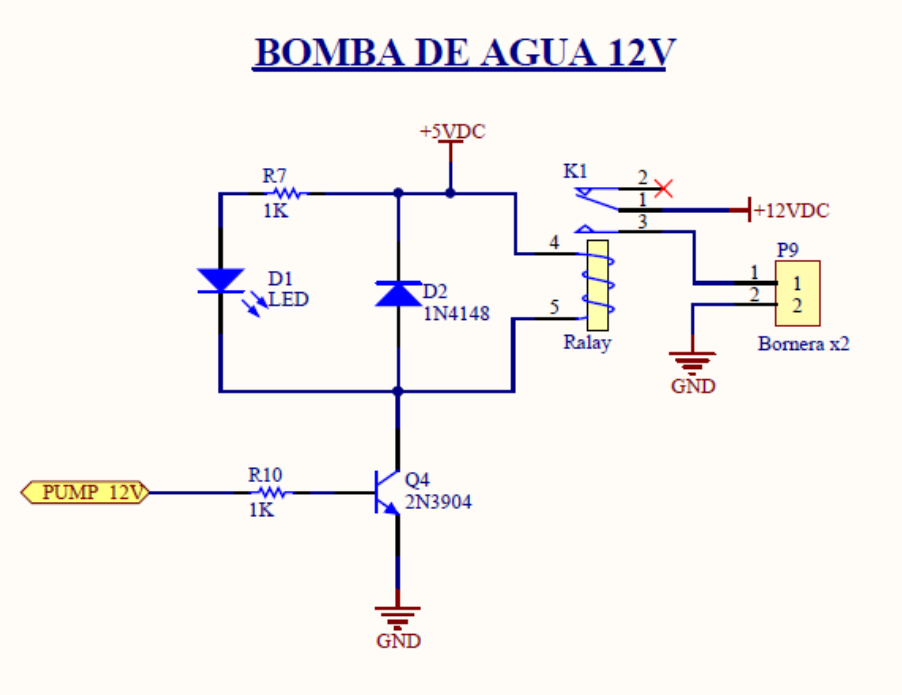


Figura 24 Conexiones de bomba de agua

En la figura 25 se observa las conexiones del ventilador extractor. La activación se la realiza mediante relay de bobina.

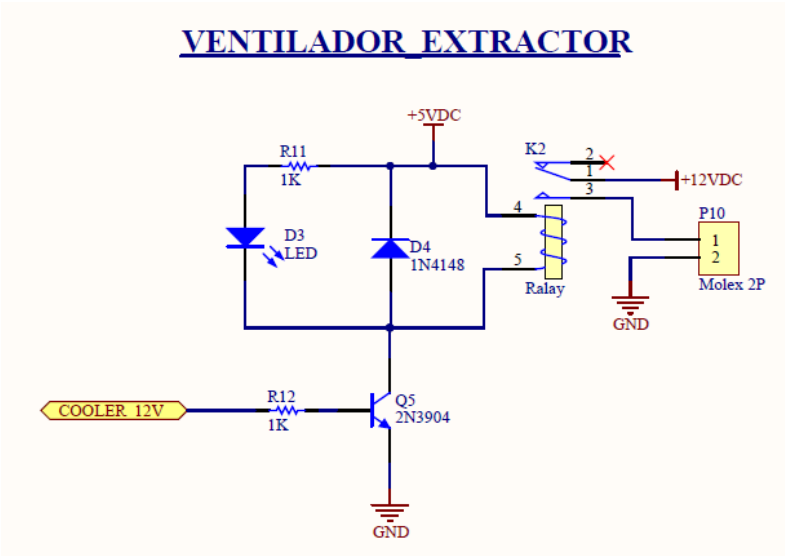


Figura 25 Conexiones de ventilador extractor

En la figura 26 se observa las conexiones del PC817X2NIP0F.

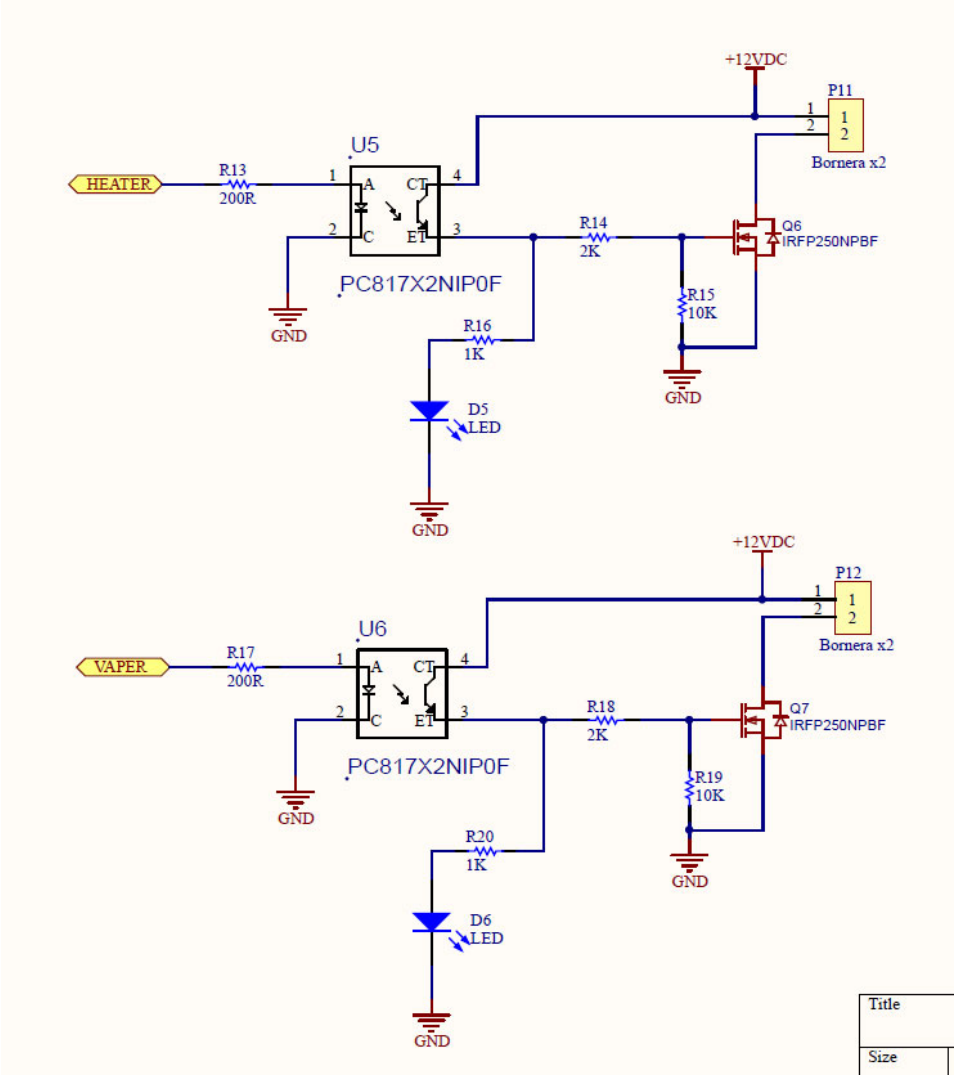


Figura 26 Conexiones PC817X2NIP0F

En la figura 27 se observa el diseño de las pistas del PCB del prototipo IoT de incubadora. El diseño impreso consta del logo de la UPS.

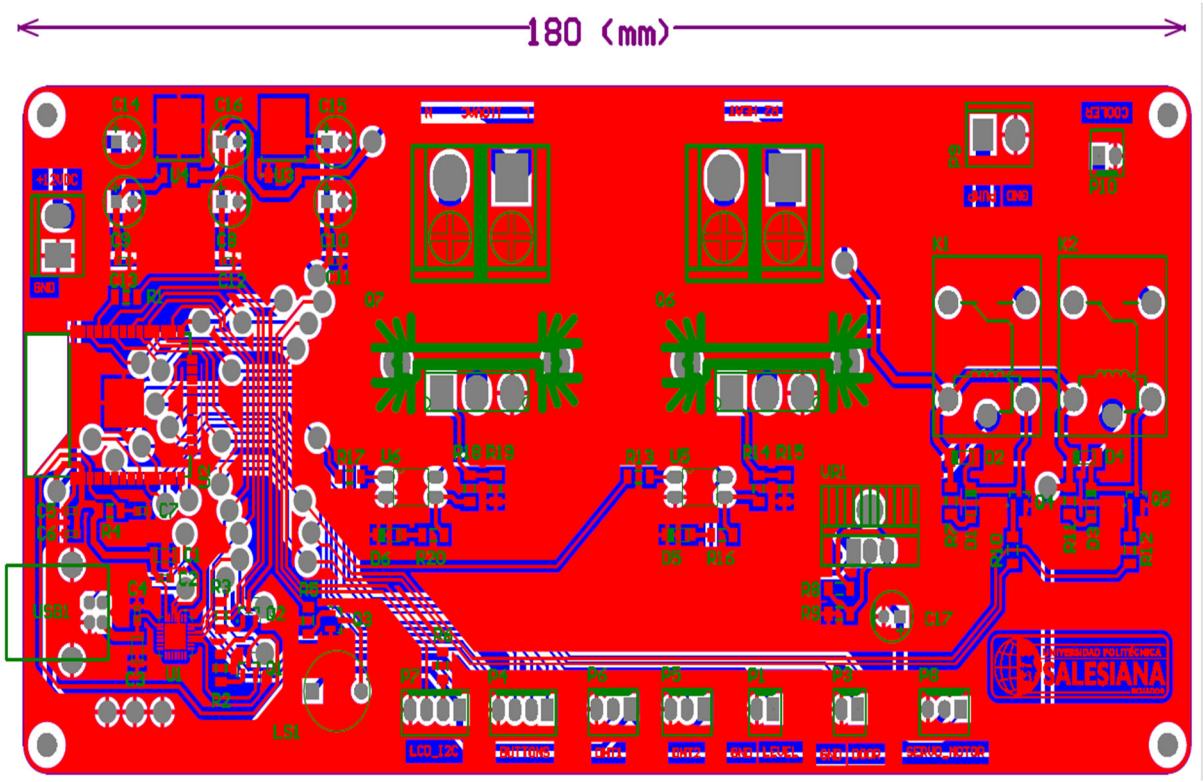


Figura 27 Diseño de pistas del PCB de prototipo IoT

En la figura 28 se observa el diseño en 3D de la placa electrónica con los elementos electrónicos. Para el diseño se realiza una correcta adecuación de los elementos electrónicos considerando las mejores recomendaciones en cuanto flujo de aire, separación, y ubicación de cada elemento electrónico.

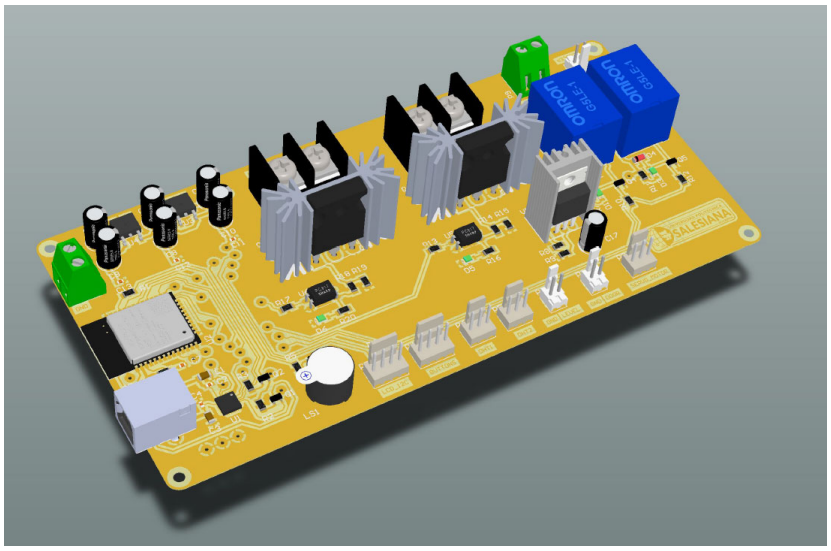


Figura 28 Diseño 3D de la placa del prototipo IoT

3.9 Diseño y ensamblaje de prototipo IoT

En la figura 29 se observa el diseño en 3D del prototipo IoT con sus elementos mecánicos y componentes electrónicos instalados. El prototipo tiene como medidas 26 cmts de altura, de largo 42 cmts y de ancho 31 cmts.

El diseño se lo realiza de forma de base rectangular para poder colocar de la mejor manera las cubetas de los huevos una encima de otra considerando el espacio de rotación que realizan estas. El diseño se realizó en base a los actuales modelos de incubadoras en 3D que existen en el mercado. Hay que considerar que tanto la fuente, la tarjeta electrónica y el rotor se encuentran fuera del case de incubación.

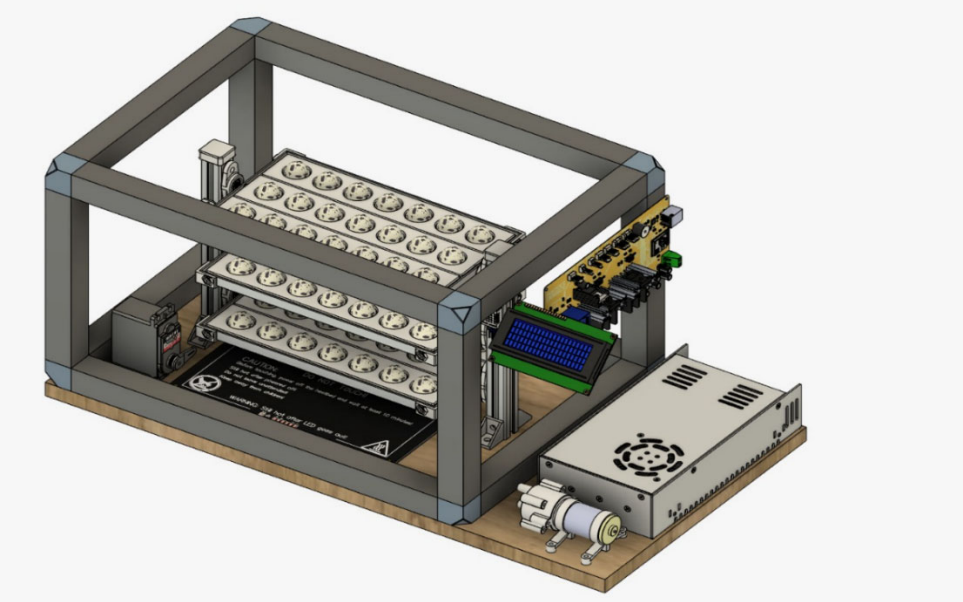


Figura 29 Diseño 3D de prototipo IoT

En las figuras del 30 al 32 se observan la placa impresa con las pistas y con los elementos electrónicos respectivamente. El diseño de la PCB se la realiza en Altium Designer.

Figura 30 Placa impresa

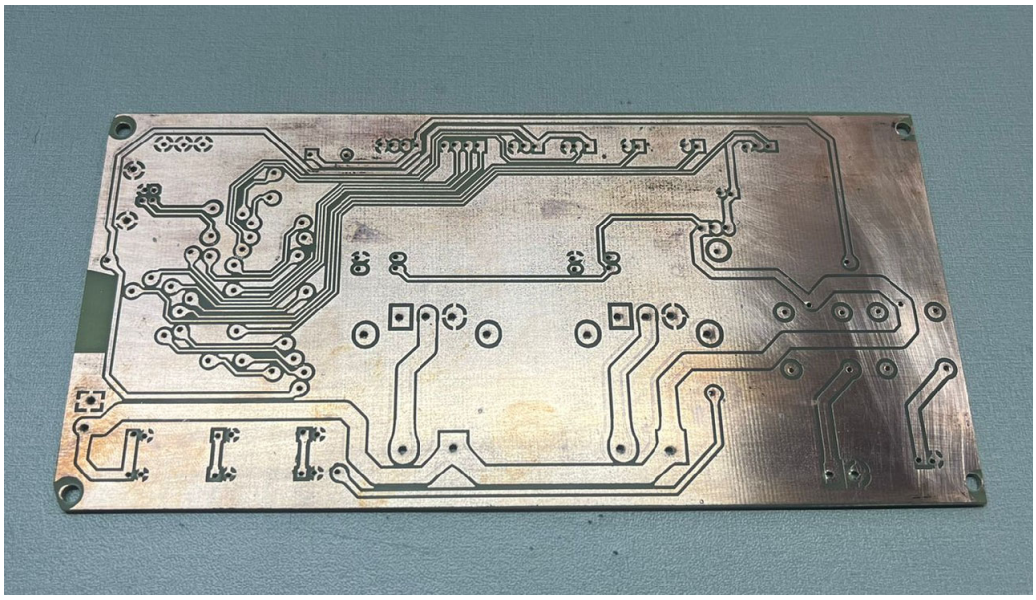


Figura 31 Pistas en placa electrónica

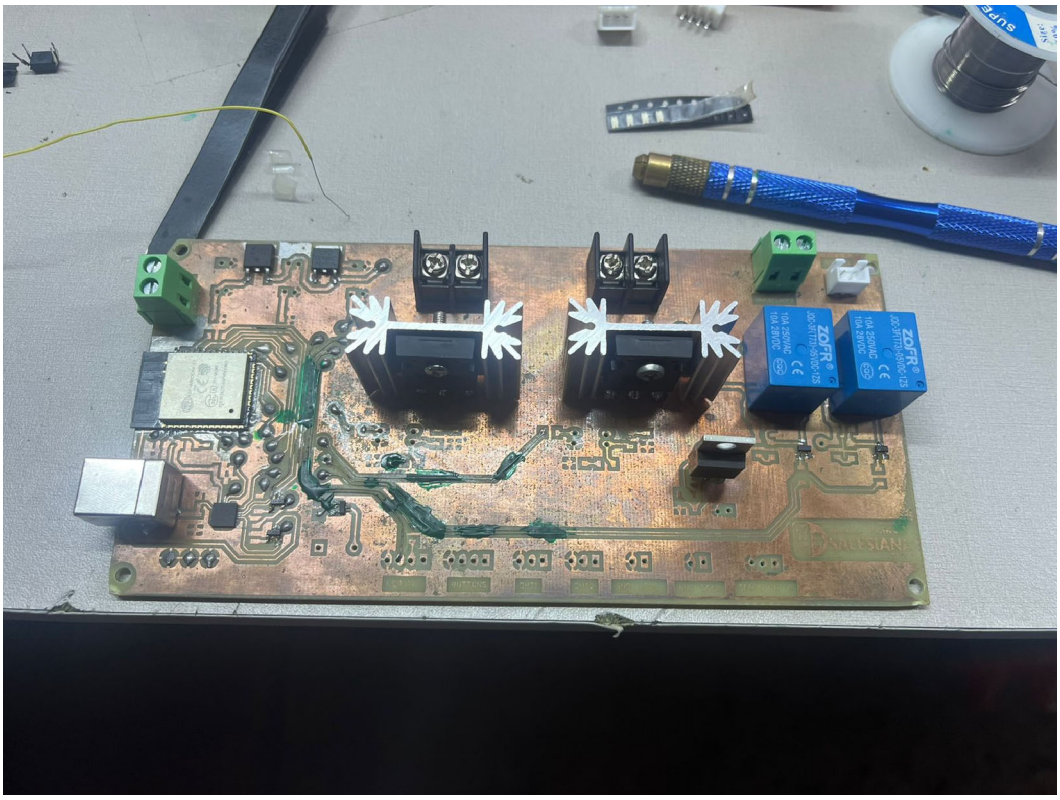


Figura 32 Soldadura de elementos electrónicos en pbc

3.10 Diseño de estructura en 3D

En las figuras del 33 al 35 se observa el diseño en 3D de la estructura del prototipo IoT. Para el diseño se utiliza el programa Fusion 360 e impresión 3D.

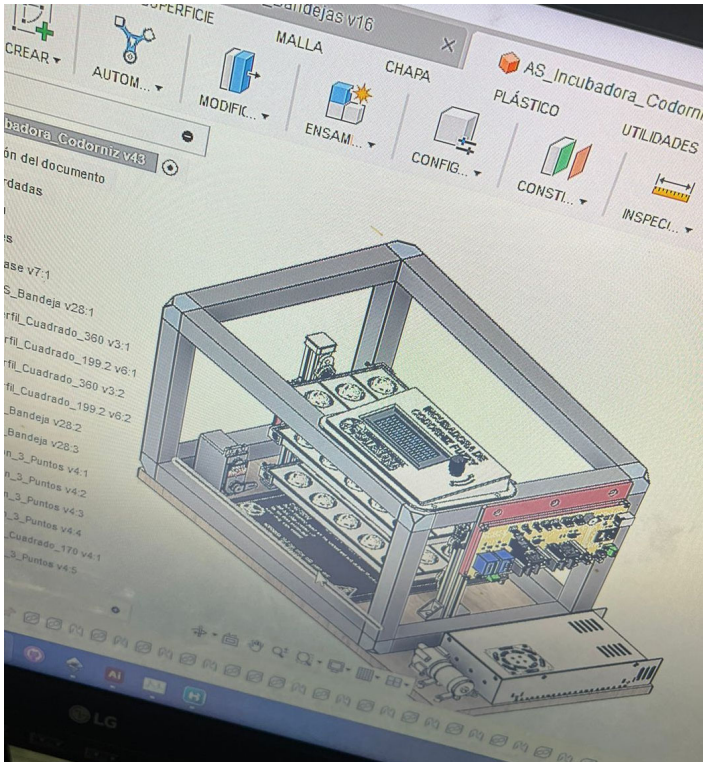


Figura 33 Diseño de prototipo IoT

Figura 34 Estructura del Prototipo IoT

Figura 35 Maqueta en 3D

En la figura 36 y 37 se observan el diseño de las bases de huevos de codorniz realizadas con impresión 3D.

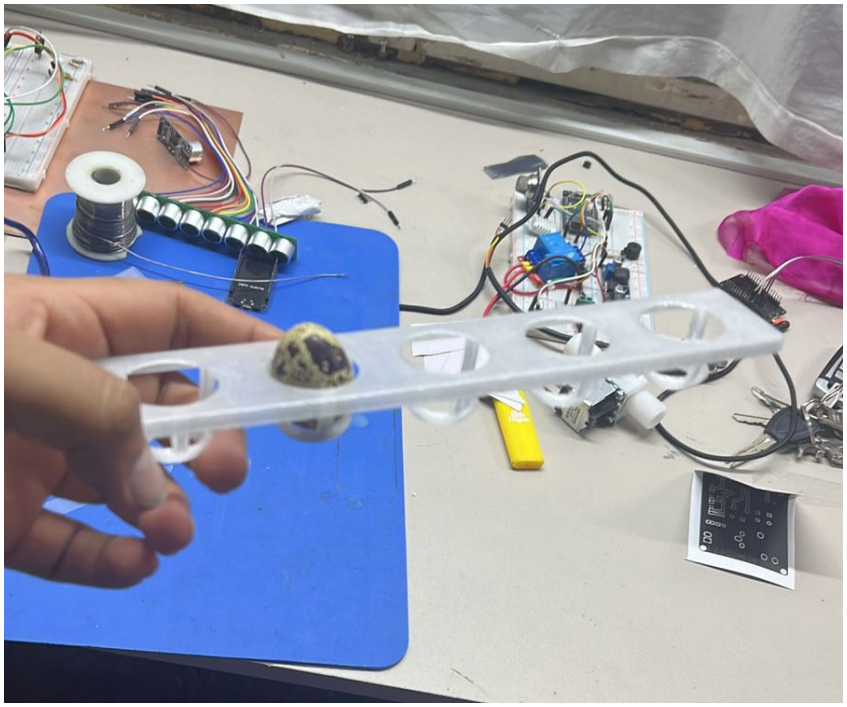


Figura 36 Bases de 3D para huevos de codorniz

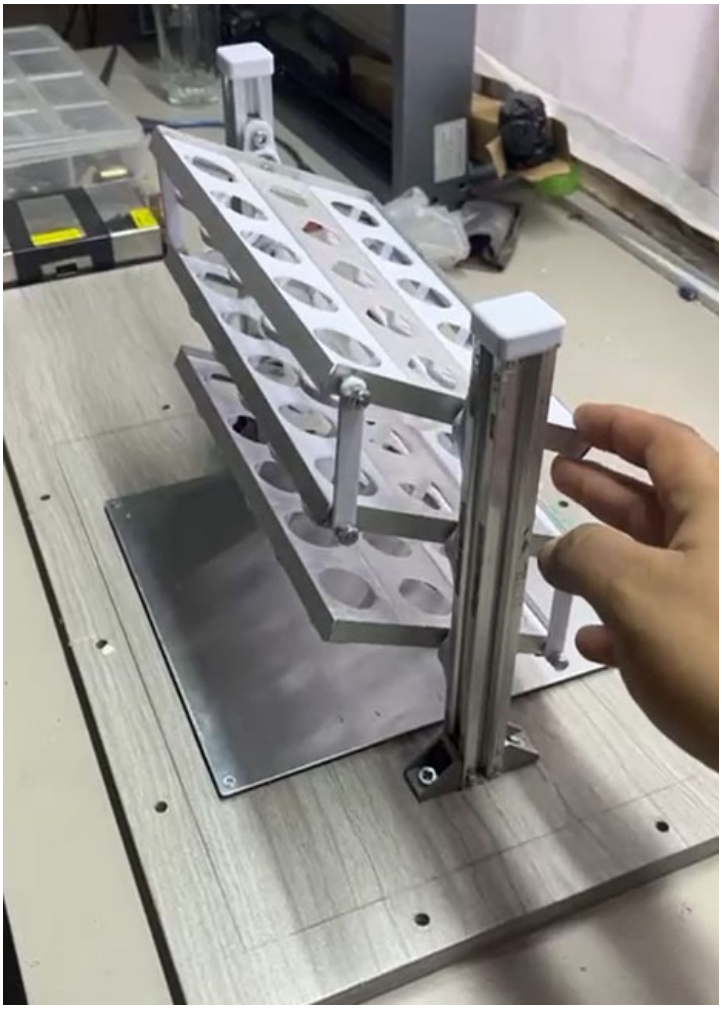


Figura 37 Segmento de rotación

En la figura 38 se observa el diseño final del prototipo IoT de incubadora de huevos de codorniz.

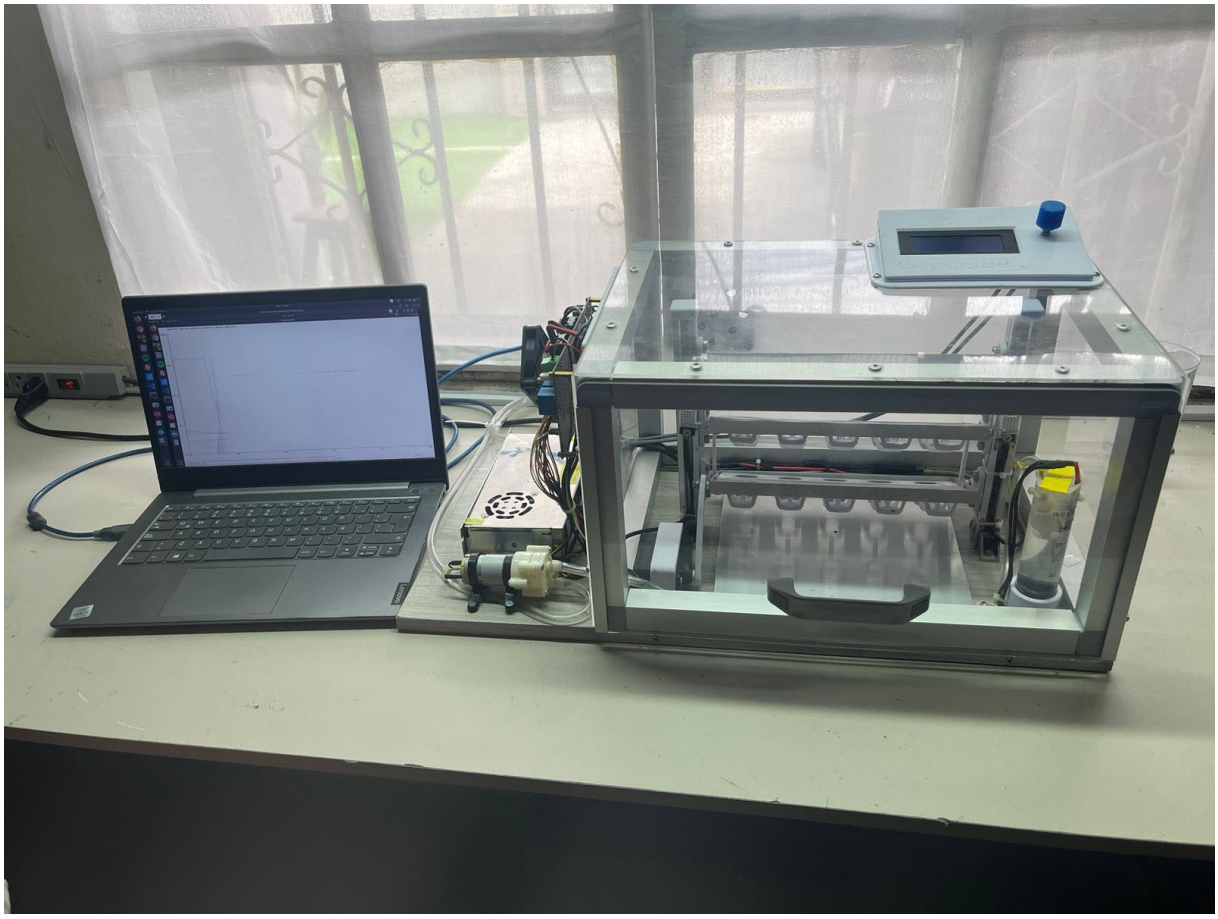


Figura 38 Diseño final de prototipo IoT de incubadora de huevos de codorniz

3.11 Configuraciones Ubidots

En esta sección se observan las configuraciones realizadas en la plataforma Ubidots, en la cual se observan los parámetros medidos con el prototipo IoT.

Inicialmente se crea la cuenta gratuita de ubidots ingresando al link https://es.ubidots.com/stem. Posteriormente en la sección de crear una cuenta gratuita se coloca nombre, apellido, email, nombre de usuario y contraseña, se señala la opción de “Mi proyecto IoT es para uso personal y no comercial”, con estos datos se crea la cuenta gratuita y llega un mail con el registro de ubidots.

Para empezar a configurar la plataforma dentro de la sesión de ubidots, se debe seleccionar el dispositivo que se usará para la toma de datos, para este caso es un ESP32.

En las figuras del 39 al 43 se observan las configuraciones de las variables y el dispositivo, asi como el historial de datos obtenidos de las variables en Ubidots.

En la figura 39 se observa el dispositivo configurado en ubidots, el cual se nombra incubadora_codorniz, al crear el dispositivo se genera la capa API, donde mostrará las variables que obtiene el dispositivo.

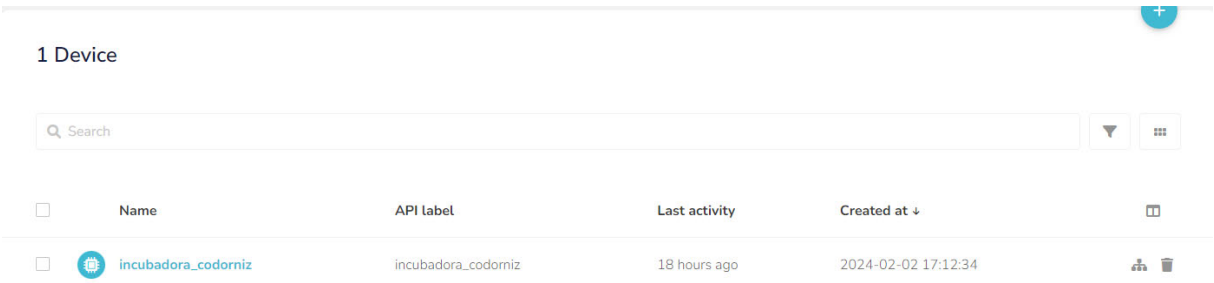


Figura 39 Configuración de dispositivo en Ubidots

En la figura 40 se observa las variables que aparecen automáticamente luego de haber agregado el dispositivo incubadora_codorniz. Estas variables son de la temperatura, humedad, rotación y luminosidad.

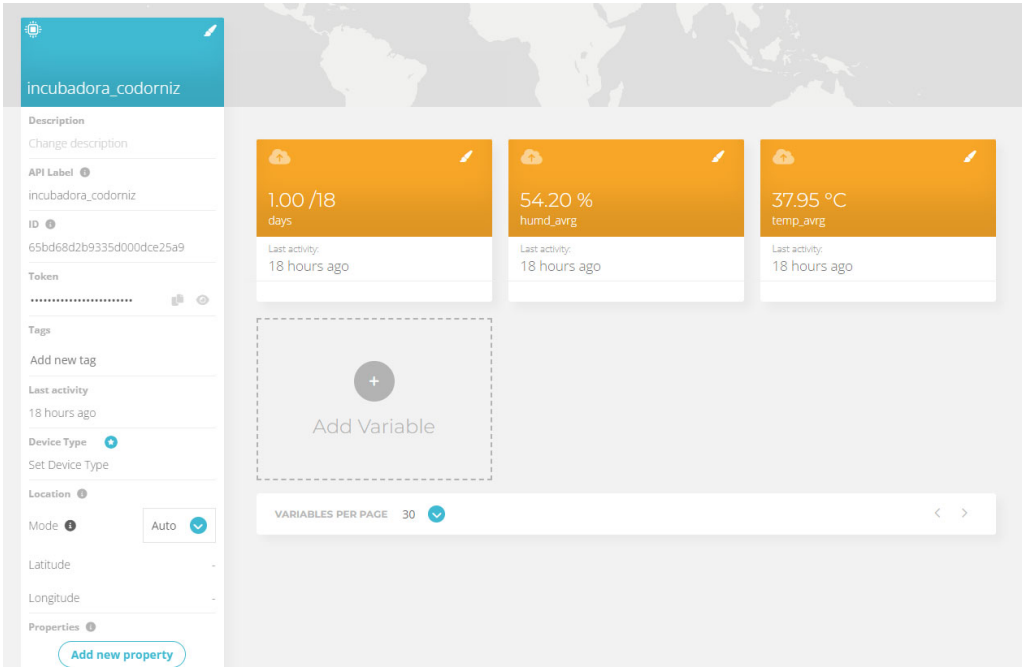


Figura 40 Configuración de variables

En la figura 41 se observan los datos del histórico de la variable día de incubación, para la muestra, el valor es de 1 día.

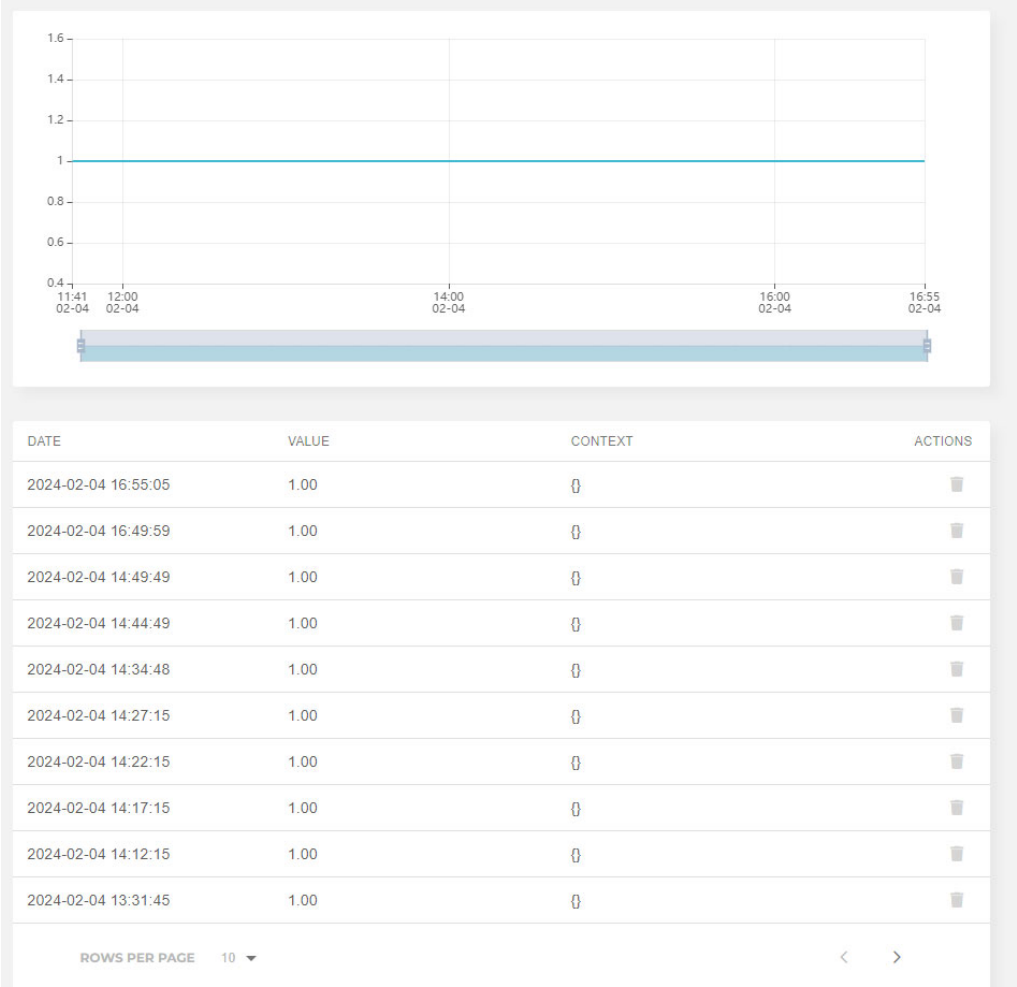


Figura 41 Histórico de variable día de incubación

En la figura 42 se observa el histórico de la variable humedad relativa.

Figura 42 Histórico de variable de humedad relativa.

En la figura 43 se observa el histórico de datos de la variable temperatura.

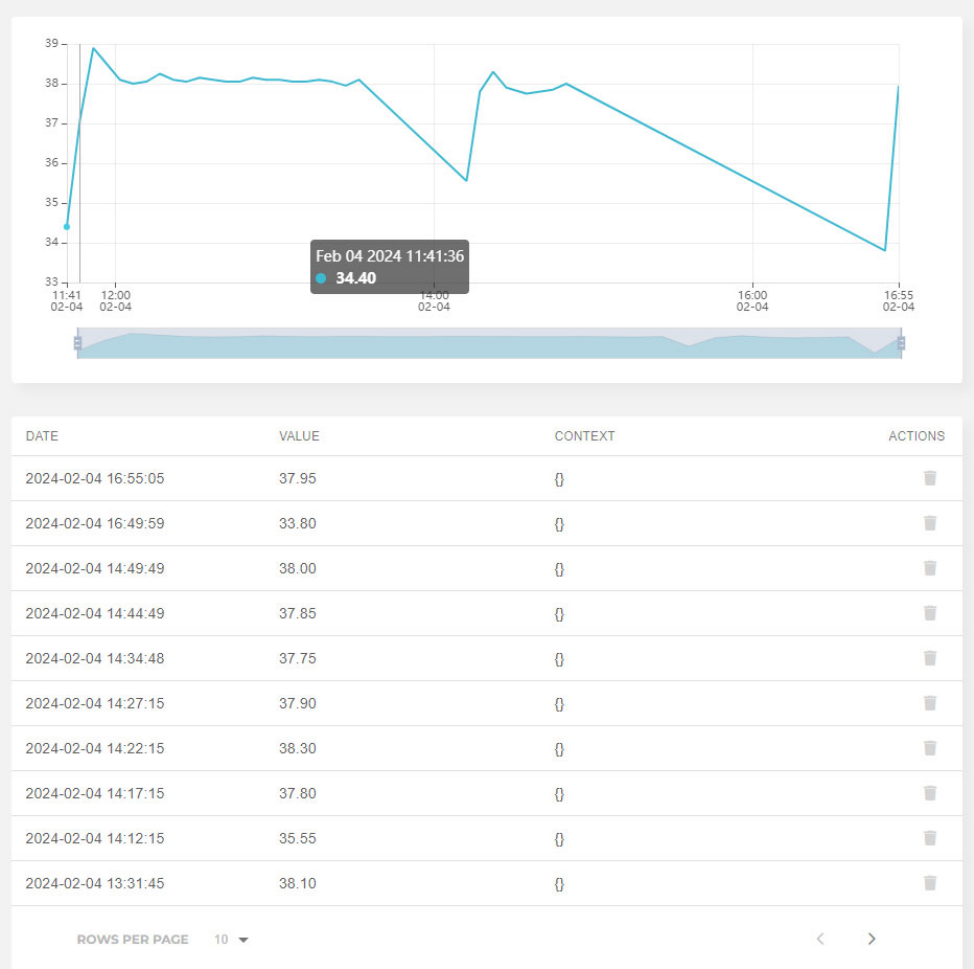


Figura 43 Histórico de variable de temperatura

4 Resultados

En esta sección se detallan los resultados obtenidos de las pruebas realizadas con el prototipo IoT de incubadora de codorniz.

En la figura 44 se observa la pantalla principal de las configuraciones realizadas con Ubidots, se puede observar además de los datos obtenidos por el prototipo, y sus variables de temperatura, humedad relativa y cantidad de días de incubación.

Al arrancar el prototipo el sensor de temperatura detecta la temperatura ambiente en el momento de las pruebas, para el clima en la ciudad de guayaquil lugar en el cual se realizaron las pruebas, la temperatura ambiente inicial es de 30,8 grados centígrados.

La humedad relativa al inicio de las pruebas es de 55%.

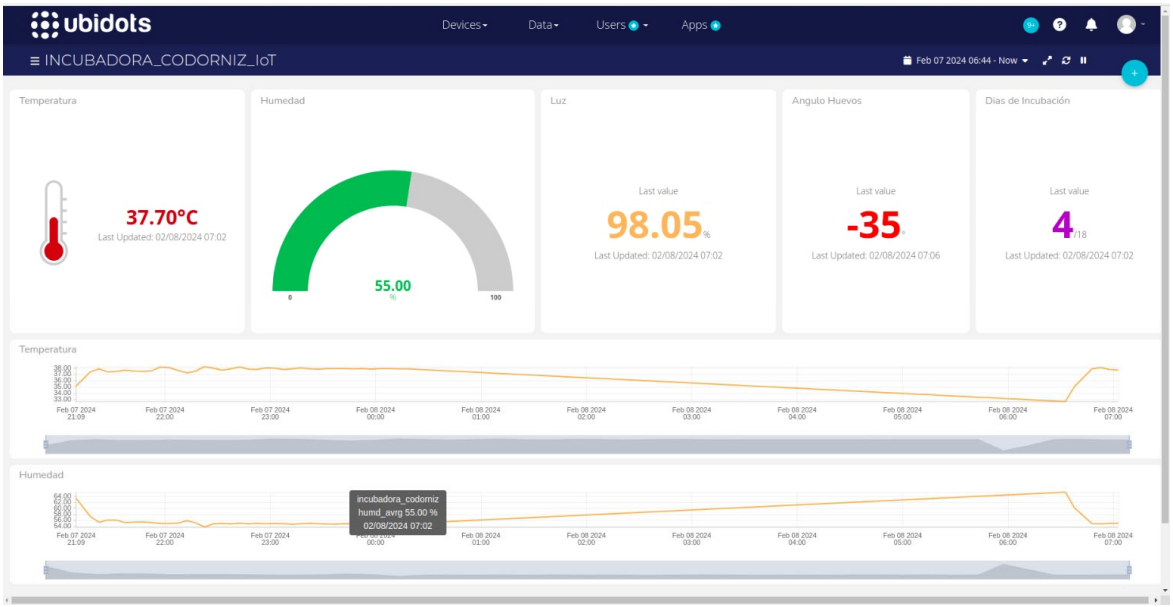


Figura 44 Pantalla principal de Ubidots

En las figuras 45 y 46 se observan los dashboards de temperatura y humedad relativa, respectivamente.

Estos datos son mostrados en la plataforma Ubidots en el momento de las pruebas de funcionamiento del prototipo.

Se utiliza un Dashboard tipo termómetro para mostrar la temperatura en tiempo real. Asi mismo se observa el dato de manera número.

Cabe indicar que la temperatura está calculada en grados Celsius.

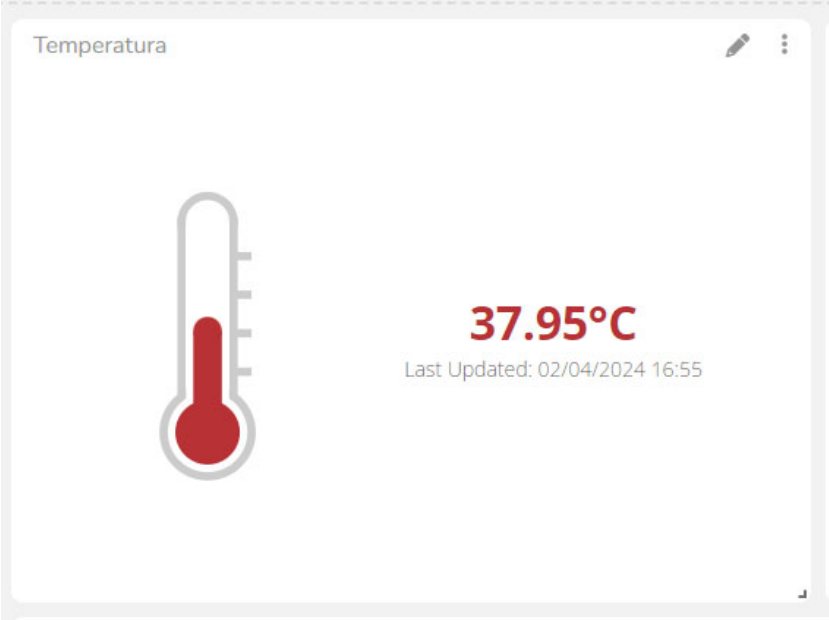


Figura 45 Dashboard de temperatura

El Dashboard de humedad relativa es tipo gauge el cual muestra con barra de color verde, el valor de la humedad relativa ambiental en %.

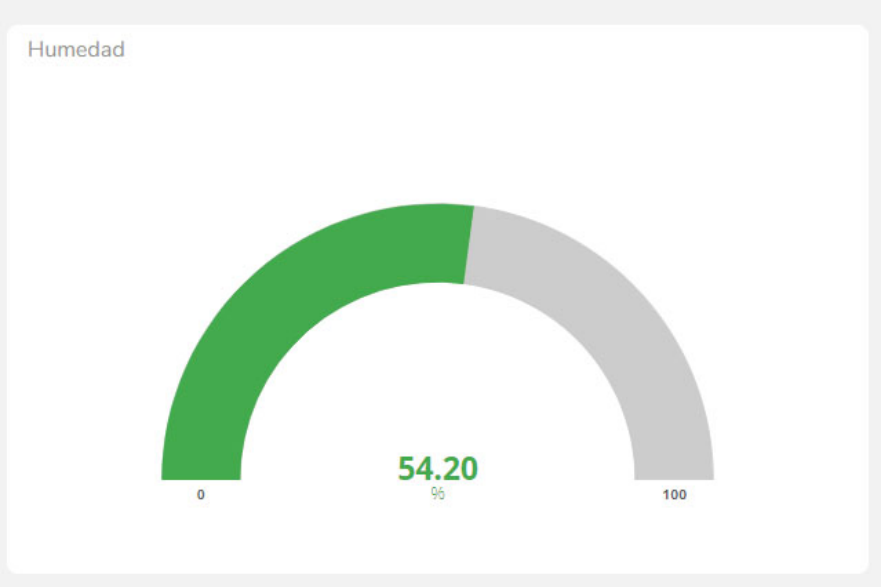


Figura 46 Dashboard de Humedad relativa

En la figura 47 se observa la cantidad de días de incubación de los huevos de codorniz.

En este Dashboard se muestra la cantidad de días que se programa para la incubación de los huevos de codornices, el cual es de 18 días.

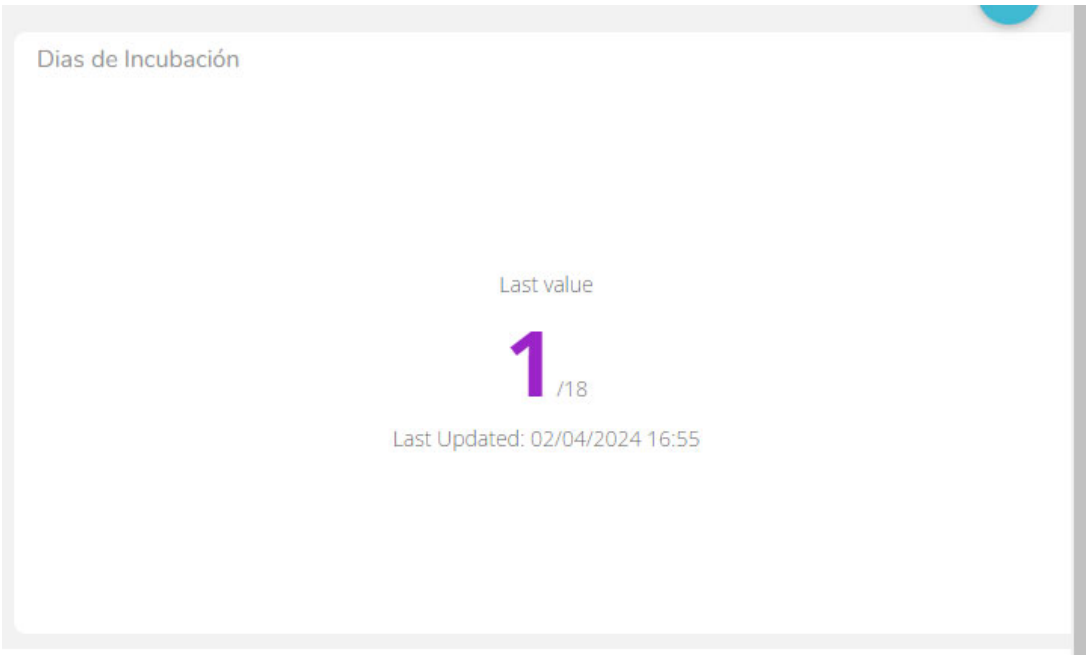


Figura 47 Cantidad de días de incubación

En la figura 48 se observa un histórico de datos de las variables de temperatura y humedad relativa.

Como dato adicional a los Dashboard se grafica un histórico de los datos tomados por los sensores de esta manera se observa la estabilización del sistema PID de temperatura a un valor adecuado para la temperatura de los huevos de codornices.

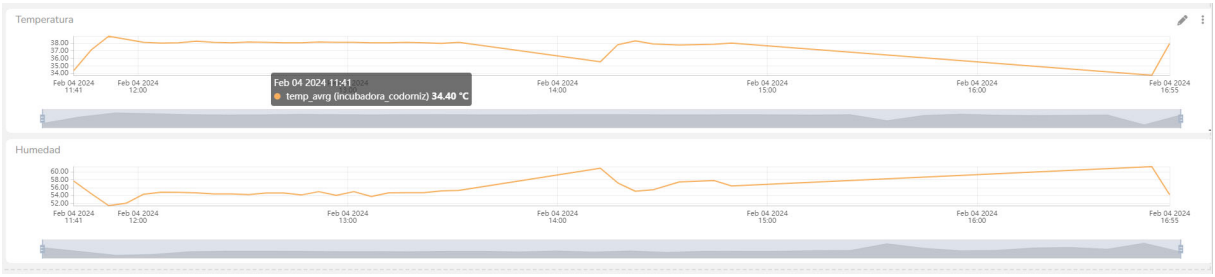


Figura 48 Histórico de temperatura y humedad relativa

En las figuras 49 y 50 se observan los datos de ángulo de inclinación y de luminosidad respectivamente.

El ángulo de inclinación se refiere al ángulo que toma la base de la incubadora de huevos.

Figura 49 Ángulo de inclinación de incubadora

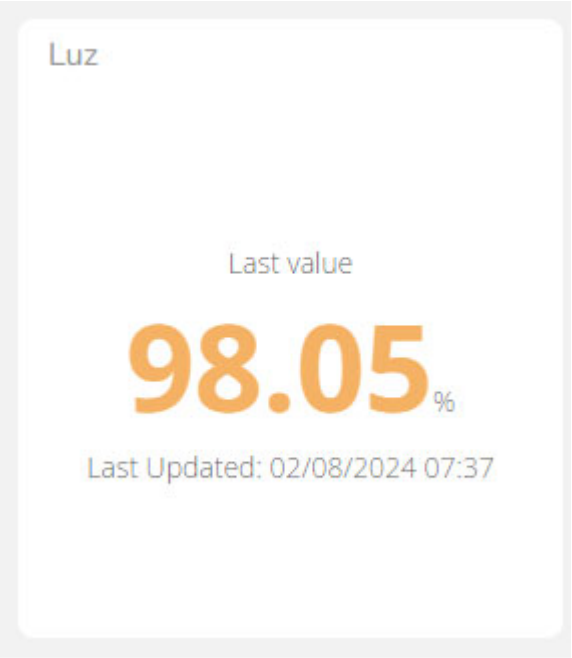


Figura 50 Luminosidad

4.1 Muestras de pruebas de funcionamiento de prototipo

En la tabla 1 se observa las muestras tomadas de los sensores, en un lapso de 5 horas durante un día. Las pruebas son tomadas en el cuarto día de incubación de los huevos entre las 21h00 y las 02h00.

Tabla 1 Datos de las pruebas de prototipo IoT

Día	Hora	Temperatura ambiente	Humedad relativa	Cantidad de luminosidad	Angulo de incubadora	Dias de incubación
8/2/2024	21:00	30,8	54%	98%	-35	4
8/2/2024	21:05	32,4	54%	98%	-35	4
8/2/2024	21:10	34,9	54%	98%	-35	4
8/2/2024	21:15	35	54%	98%	-35	4
8/2/2024	21:20	35,2	54%	98%	-35	4
8/2/2024	21:25	35,2	54%	98%	-35	4
8/2/2024	21:30	35,2	54%	98%	-35	4
8/2/2024	21:35	35,3	54%	98%	-35	4
8/2/2024	21:40	35,4	54%	98%	-35	4
8/2/2024	21:45	35,5	54%	98%	-35	4
8/2/2024	21:50	35,6	54%	98%	-35	4
8/2/2024	21:55	35,7	54%	98%	-35	4
8/2/2024	22:00	35,8	54%	98%	-35	4
8/2/2024	22:05	35,9	54%	98%	-35	4
8/2/2024	22:10	36	54%	98%	-35	4
8/2/2024	22:15	36,1	54%	98%	-35	4
8/2/2024	22:20	36,2	54%	98%	-35	4
8/2/2024	22:25	36,3	54%	98%	-35	4
8/2/2024	22:30	36,4	54%	98%	-35	4
8/2/2024	22:35	36,5	54%	98%	-35	4
8/2/2024	22:40	36,6	55%	98%	-35	4
8/2/2024	22:45	36,7	55%	98%	-35	4
8/2/2024	22:50	36,8	55%	98%	-35	4
8/2/2024	22:55	36,9	55%	98%	-35	4
8/2/2024	23:00	37	55%	98%	-35	4
8/2/2024	23:05	37,1	55%	98%	-35	4
8/2/2024	23:10	37,2	55%	98%	-35	4
8/2/2024	23:15	37,3	55%	98%	-35	4
8/2/2024	23:20	37,4	55%	98%	-35	4
8/2/2024	23:25	37,5	55%	98%	-35	4
8/2/2024	23:30	37,6	55%	98%	-35	4
8/2/2024	23:35	37,7	55%	98%	-35	4
8/2/2024	23:40	37,8	55%	98%	-35	4
8/2/2024	23:45	37,9	55%	98%	-35	4
8/2/2024	23:50	38	55%	98%	-35	4
8/2/2024	23:55	38	55%	98%	-35	4

8/2/2024	0:00	38	55%	98%	-35	4
8/2/2024	0:05	38	55%	98%	-35	4
8/2/2024	0:10	38	55%	98%	-35	4
8/2/2024	0:15	38	55%	98%	-35	4
8/2/2024	0:20	38	55%	98%	-35	4
9/2/2024	0:25	38	55%	98%	-35	4
10/2/2024	0:30	38	55%	98%	-35	4
11/2/2024	0:35	38	55%	98%	-35	4
12/2/2024	0:40	38	55%	98%	-35	4
13/2/2024	0:45	38	55%	98%	-35	4
14/2/2024	0:50	38	55%	98%	-35	4
15/2/2024	0:55	38	55%	98%	-35	4
16/2/2024	1:00	38	55%	98%	-35	4
17/2/2024	1:05	38	55%	98%	-35	4
18/2/2024	1:10	38	55%	98%	-35	4
19/2/2024	1:15	38	55%	98%	-35	4
20/2/2024	1:20	37	55%	98%	-35	4
21/2/2024	1:25	36	55%	98%	-35	4
22/2/2024	1:30	35	55%	98%	-35	4
23/2/2024	1:35	34	55%	98%	-35	4
24/2/2024	1:40	33	55%	98%	-35	4
25/2/2024	1:45	32	55%	98%	-35	4
26/2/2024	1:50	31	55%	98%	-35	4

27/2/2024	1:55	30	55%	98%	-35	4
28/2/2024	2:00	29	55%	98%	-35	4

En las figuras 51, 52 y 53 se observan las gráficas de los datos obtenidos durante el muestreo.

Se puede observar la estabilización de la temperatura gracias al control realizado con la lógica de la librería PID_fuzzy quemada en el esp32. Así como también se observa la humedad relativa estabilizada.

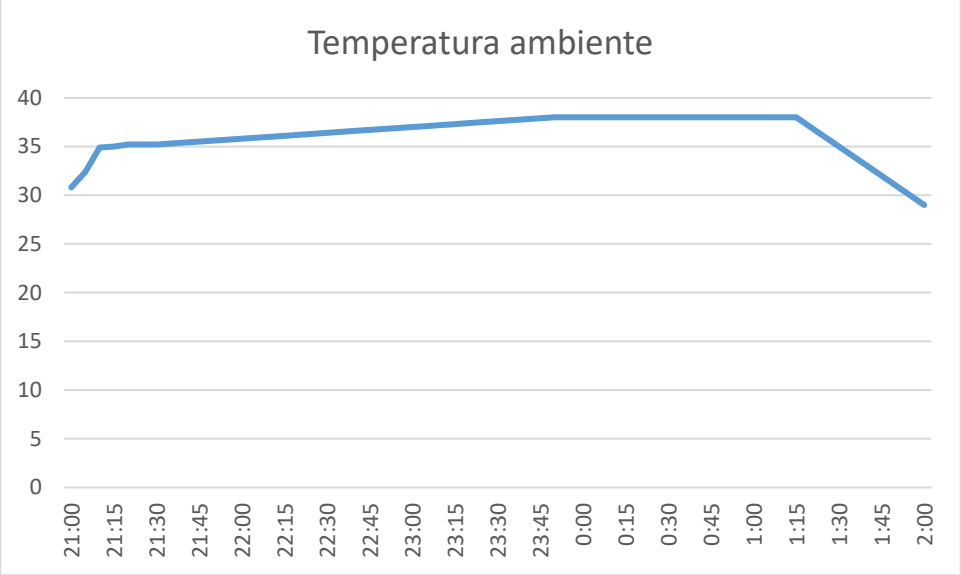


Figura 51 Datos de Temperatura ambiente

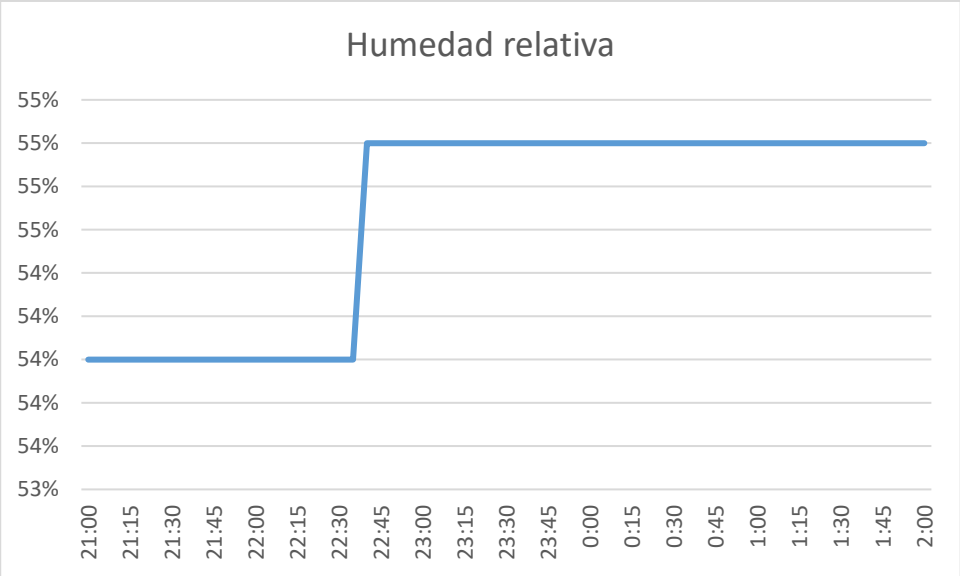


Figura 52 Datos de Humedad relativa

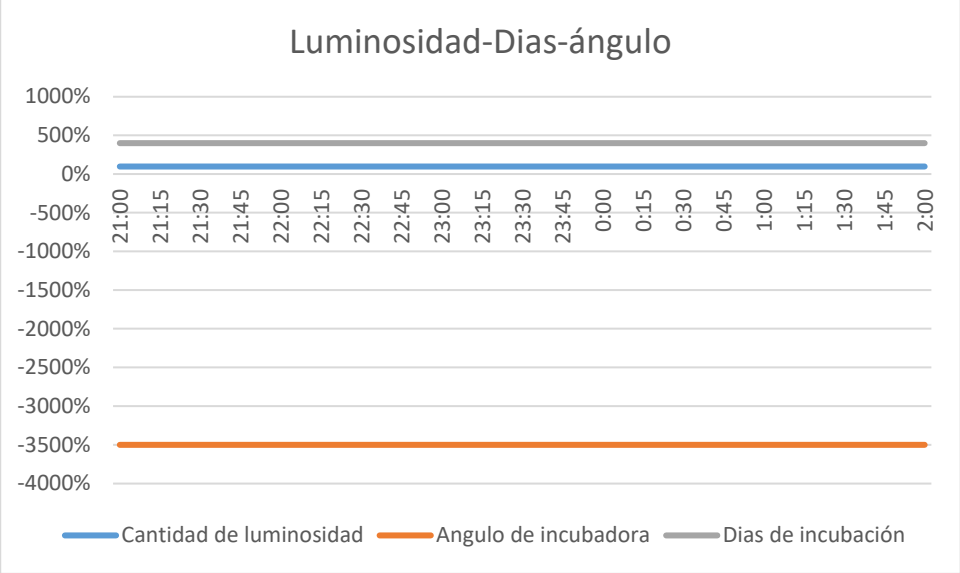


Figura 53 Datos de luminosidad, días de incubación, ángulo de inclinación

4.2 Configuraciones y resultados de alertas email

En esta sección se detalla los resultados de las configuraciones realizadas con las alertas vía mail de los datos obtenidos por el prototipo. Tal como se observan las figuras del 54 al 59.

Las configuraciones de alerta vía mail ayudan a realizar un control del funcionamiento del sistema desarrollado en este trabajo, con el mismo se configuran datos de umbral adecuados para la incubadora de huevos.

En la figura 54 se observa la creación del evento, se selecciona la variable incubadora_codorniz, la condición hace referencia a que se activará el evento cuando el dato o trigger sea mayor o igual que 16 para este caso.

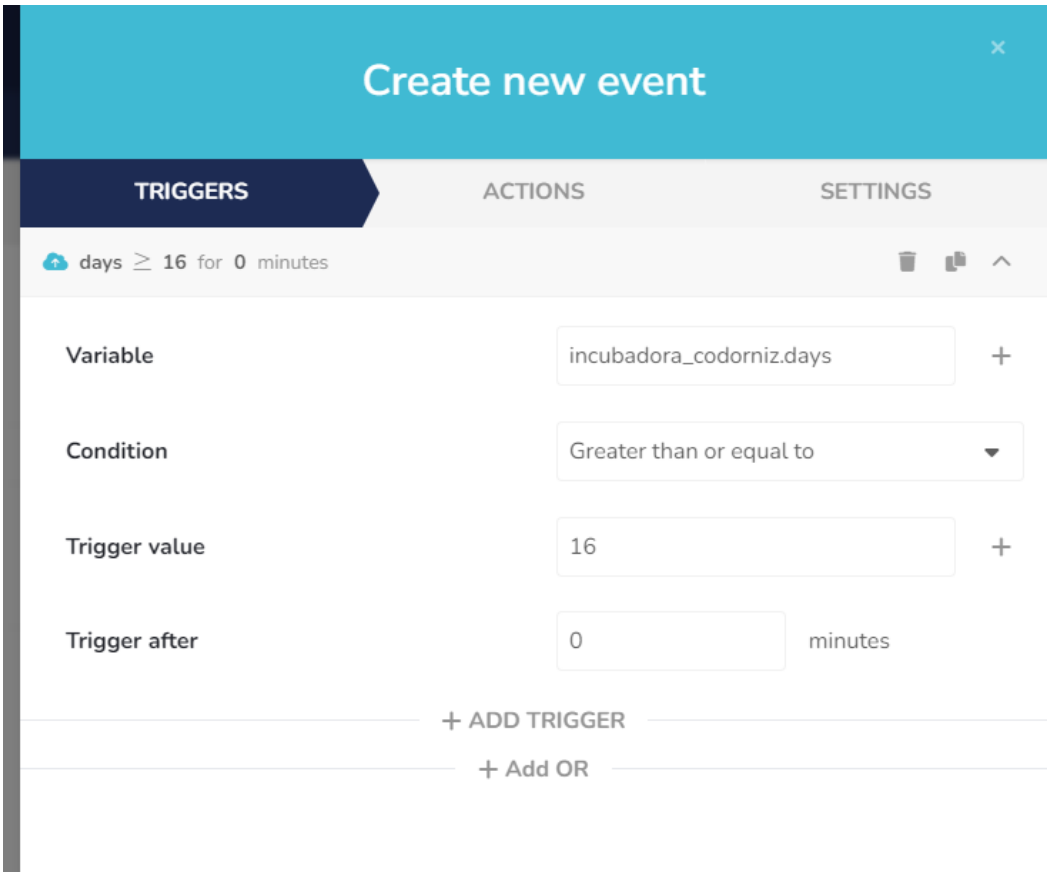


Figura 54 Evento de días de incubación

En la figura 55 se observa la configuración del envío de email, se coloca el título del correo o subject, el mensaje y se configura la repetición, para este caso se envía 3 correos cada 5 minutos.

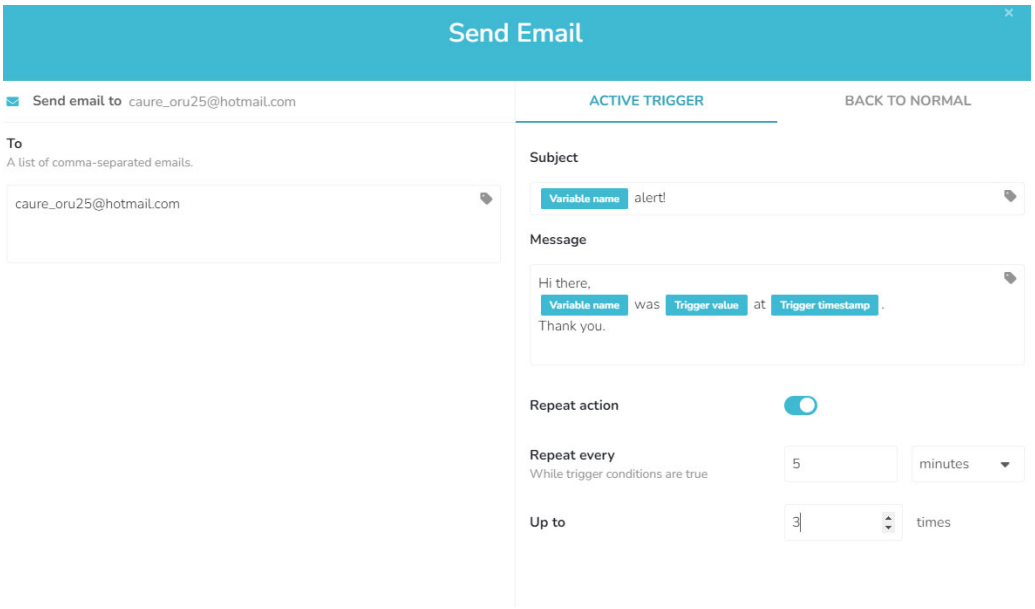


Figura 55 Configuración de envio de mail

Finalmente en la figura 56 se observa como se setea el nombre de la alerta.

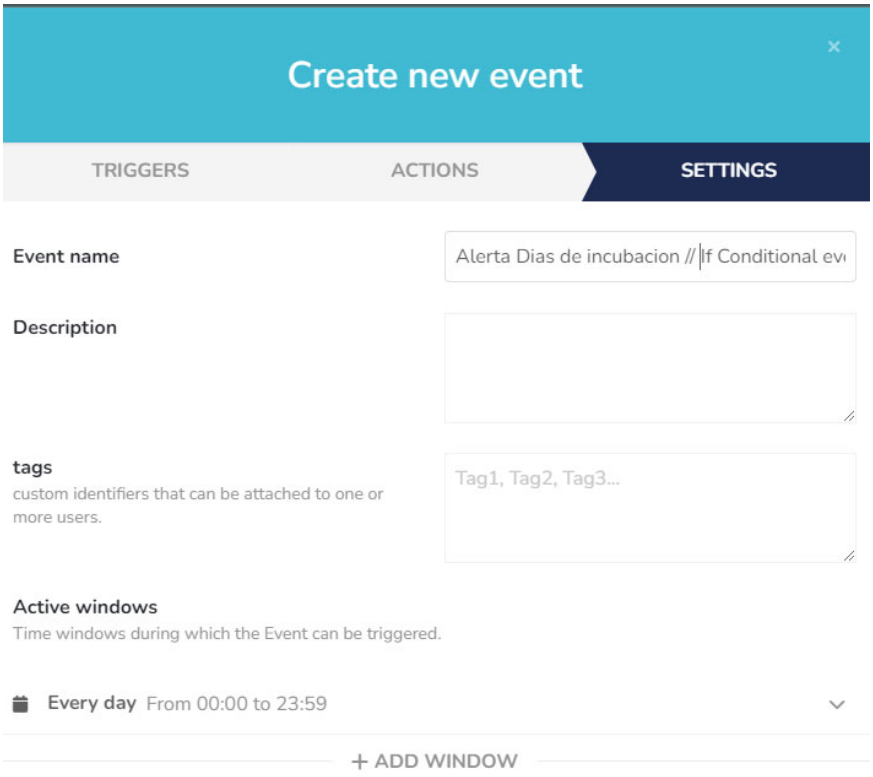


Figura 56 Nombre de alerta

En la figura 57 se observa la configuración de la alerta de humedad relativa.

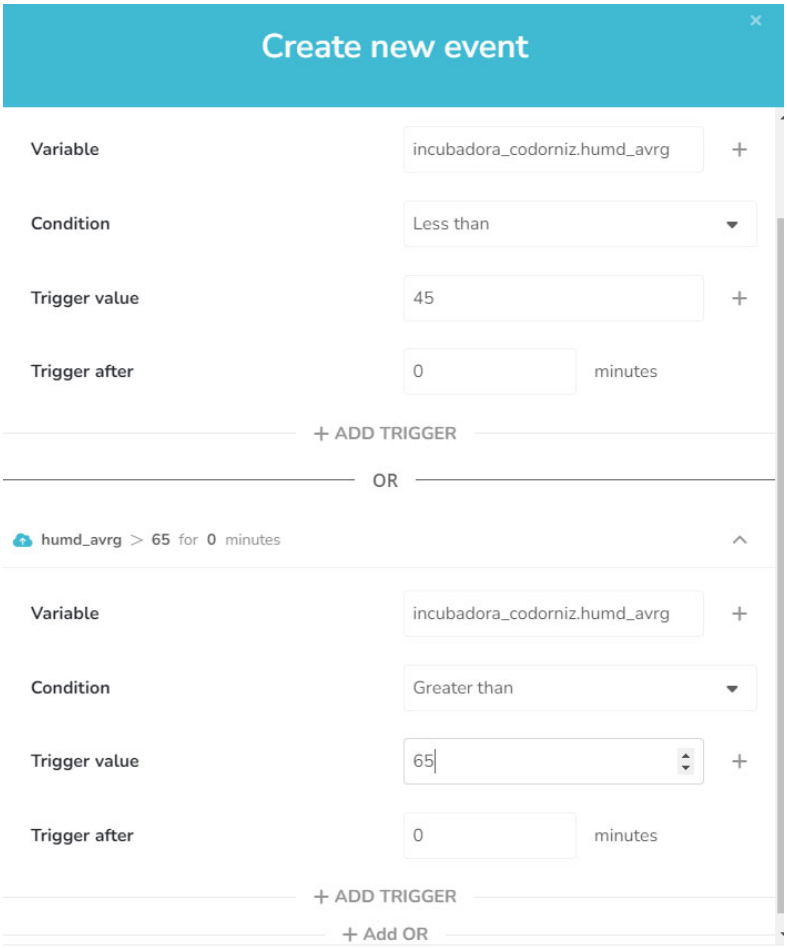


Figura 57 Evento de humedad relativa

En la figura 58 se observa la configuración detallada de la variable temperatura.

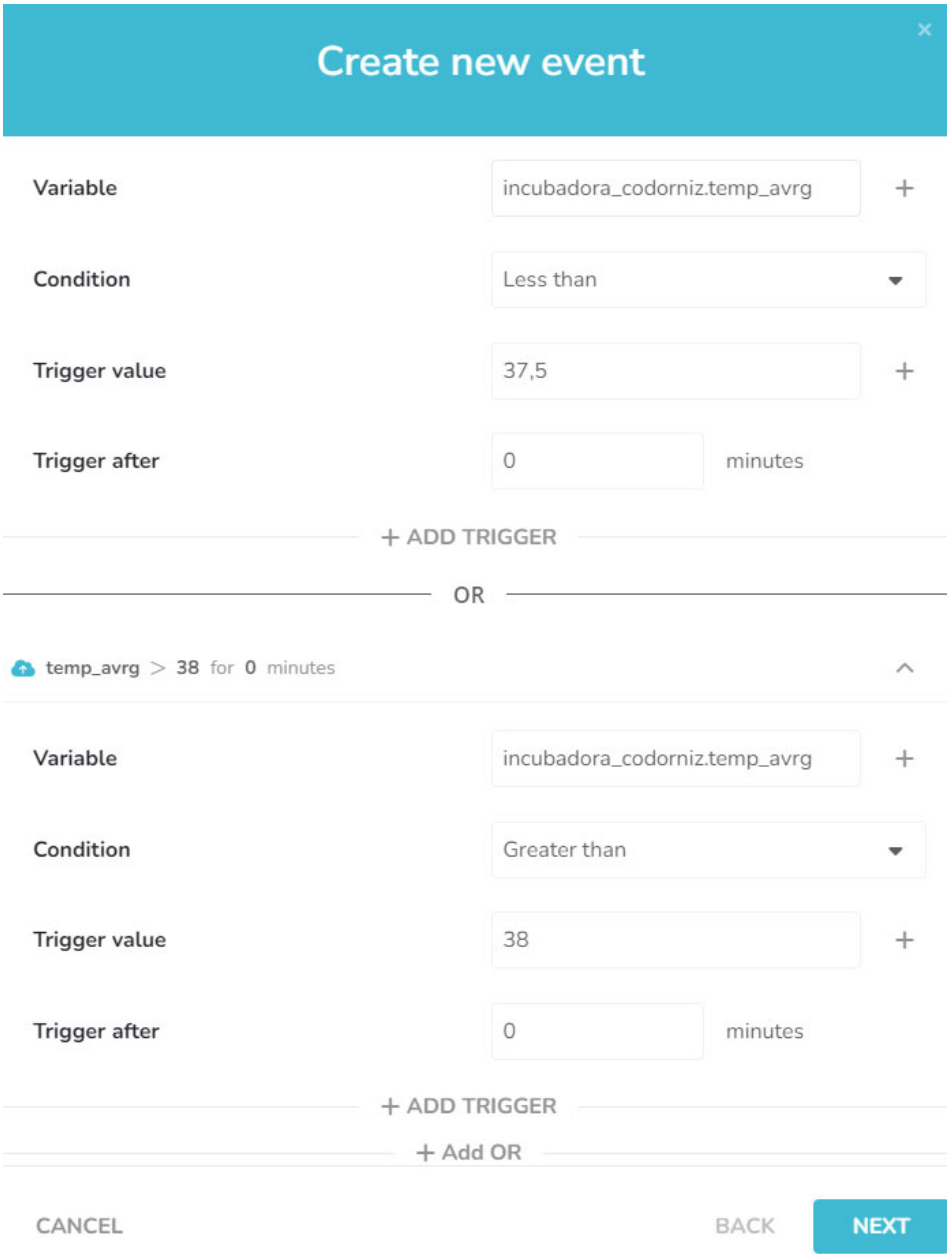


Figura 58 Evento de temperatura

Al termino de las configuraciones se observa en el panel principal, las alertas vía email realizadas. Ver figura 59.

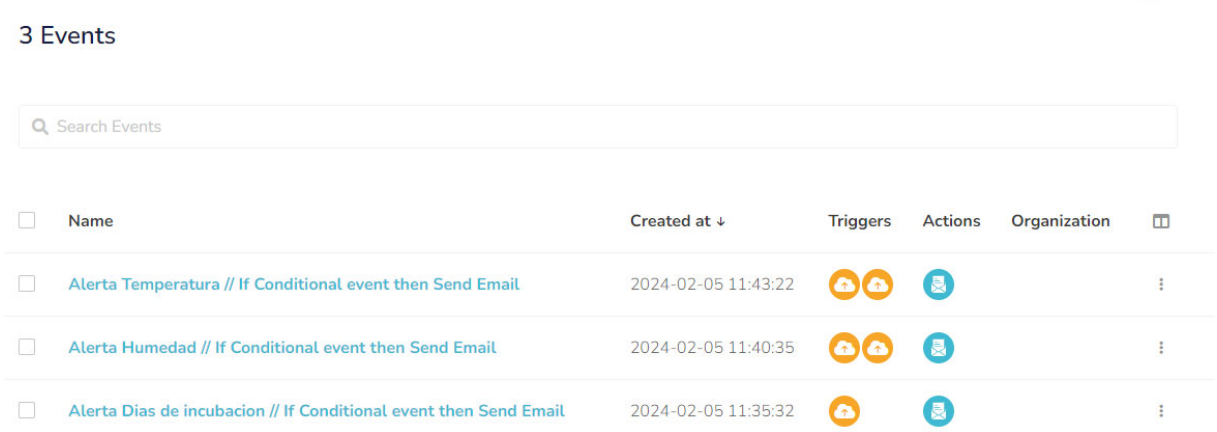


Figura 59 Configuraciones de eventos

4.3 Resultados del controlador PID

En esta sección se muestran las gráficas de los resultados del controlador PID de temperatura y humedad relativa. Tal como se observa en las figuras 60 y 61.

Las gráficas son tomadas del monitor del IDE de Arduino, el cual demuestran como se estabiliza la temperatura gracias al código de programación del ESP32 que trabaja con la librería PID_Fuzz.h, el código de error se afina visualizando los datos obtenidos durante las pruebas.

En la observación de las gráficas se determina que la estabilización se la realiza aproximadamente en los 35 grados para la variable de temperatura.

Se puede observar en la figura 60, los valores de set_point-h (humedad), se definen como color azul en la gráfica, los valores de entrada de datos de humedad de color rojo, valores de salida de humedad color verde, el set_point_t hace referencia al dato de temperatura y en la gráfica se representa de color naranja.

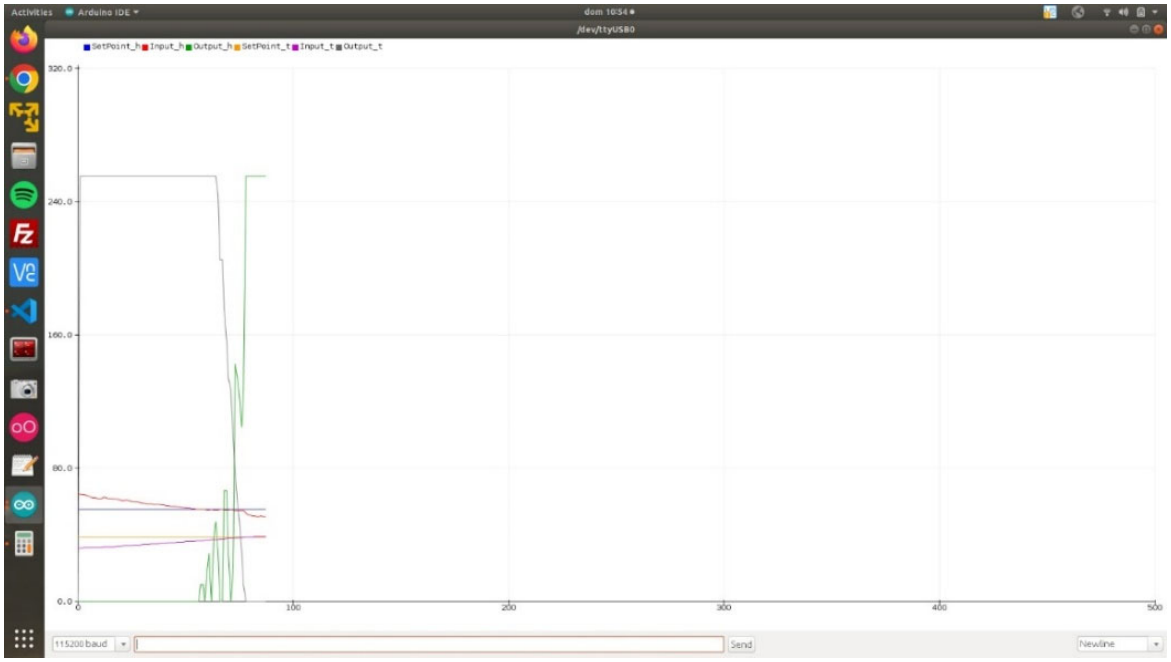


Figura 60 Resultados del controlador PID_fuzzy

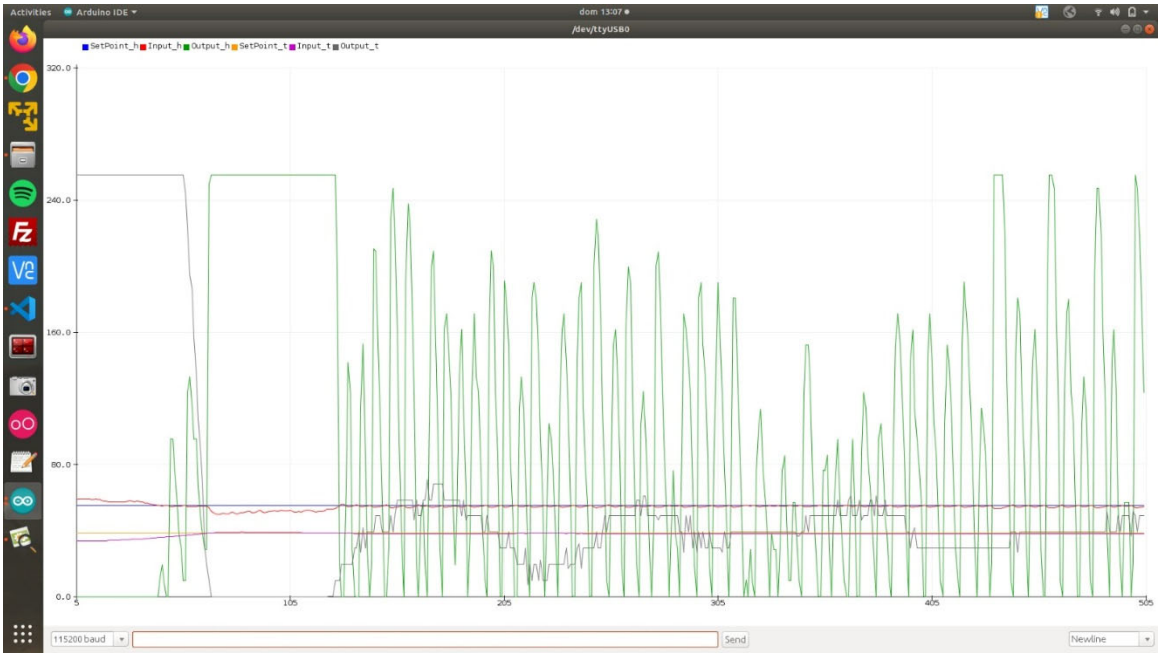


Figura 61 Mediciones y resultados del controlador PID

Con esta gráfica se demuestra la estabilización de la temperatura y humedad relativa dentro de la incubadora de huevos de codorniz.

Así mismo ayuda a realizar ajustes al código de programación del ESP32 en caso de que estos valores no se estabilicen.

La afinación o tuning se la aplica en el código de la librería PID_Fuzzy. Y todos los datos observables son registrados en el histórico de las variables en Ubidots.

5 Conclusiones

Las conclusiones de este trabajo de investigación son las siguientes:

- De acuerdo con la investigación de diferentes tipos de incubadora de huevos que existen en el mercado, se elaboró un prototipo de incubación artificial utilizando control difuso y tecnologías IoT para ser aplicado en huevos de codorniz, considerando abaratar costos y utilizar materiales de buena calidad específicamente en la impresión 3D. El diseño electrónico se lo realizó utilizando software Altium Designer para la PCB, este prototipo inicial tiene la ventaja de los diseños electrónicos que en futuro pueden utilizarse para mejoras en caso de que se necesite colocar mas cubetas de huevos de codorniz en una sola caja.

- Mediante el uso de software para impresión en 3D Fusion 360, se realizó el diseño considerando las medidas adecuadas de acuerdo con el tamaño de los huevos y así determinar la cantidad de cubetas del prototipo. También se utilizó una estructura de aluminio para el soporte principal de las piezas en 3D. Se considera que el prototipo no encarezca su valor por el uso de muchos materiales.

- Se realiza la estabilización de la temperatura y la humedad relativa dentro de la incubadora de huevos de codorniz aplicando lógica difusa fuzzy en el código de programación del ESP32, mediante la librería PID_Fuzzy diseñada para este proyecto de investigación. Este código controla el encendido y apagado de la resistencia que hará que la temperatura se mantenga en 38 grados de manera estable así como mantener la humedad relativa en 54%, para esto previamente se definieron los valores de set_point dentro del código del ESP32.

- Para el monitoreo remoto de los diversos parámetros de temperatura, humedad relativa, rotación de cubeta y luminosidad del prototipo de incubadora de huevos de codorniz se utilizó la plataforma Ubidots con una cuenta stem gratuita que permite monitorizar en el histórico de datos y observar gráficamente el comportamiento de las variables, esta plataforma permite el envío de alertas vía email, lo cual es de suma importancia para este estudio.

6 Recomendaciones

Como recomendaciones se tienen las siguientes.

- El prototipo IoT de incubadora de huevos de codorniz puede ser utilizado también para incubación de huevos de gallina o de otras aves, lo que se deberá cambiar de ser el caso, serían las estructuras en 3D que reposan los huevos, de ahí el principio es el mismo, monitorizar temperatura, humedad relativa, rotación de huevos para una correcta incubación.

- La plataforma en la nube utilizada para este proyecto es Ubidots, se puede utilizar otro tipo de plataformas como ThinkSpeak o Grafana Cloud para el monitoreo remoto, así como tambien para las configuraciones de las alertas. Considerar que Ubidots tiene una plataforma gratuita pero limitada, ya que no permite descargar el histórico de datos de las variables, si se requiere de este, se debe cancelar un valor elevado, por esto se recomienda que el prototipo tenga una base de datos local de respaldo como una Raspberry PI que almacene la data en la microSD y posteriormente pueda ser descargado remotamente, para esto se puede utilizar Grafana local que es una aplicación gratuita que debe ser instalada como servidor en la Raspberry PI.

- El diseño de este proyecto IoT está realizado para 45 huevos de codorniz. Para tener más capacidad de incubación se debe realizar un rediseño que implique más sensores y más capacidad para el soporte de los huevos.

- Este proyecto de titulación sirve para realizar prácticas relacionadas a tecnologías IoT, es recomendable que se replique más prototipos de esta índole para el estudio en controladores PID.

7 Bibliografía

3ds.com. (2024). Impresión 3D, ¿qué es y cómo funciona? | Dassault Systèmes®. https://www.3ds.com/es/make/guide/process/3d-printing

bbva.ch. (2024). La industria de la impresión 3D, un sector en crecimiento. https://www.bbva.ch/noticia/la-industria-de-la-impresion-3d-un-sector-en-crecimiento/

Bello, E. (2021). Lógica Difusa o Fuzzy Logic: Qué es y cómo funciona + Ejemplos. Thinking for Innovation. https://www.iebschool.com/blog/fuzzy-logic-que-es-big-data/

Bitcu.co. (2024). Ubidots: La plataforma IoT enfocada al manejo de datos – BitCuco. https://bitcu.co/ubidots/

Chen, G., & Tat Pham, T. (2024). Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems Introduction to.

Cruz, K. L., & Vera, F. F. (2021). Temperature control system for a hatchery. Tekhnê, 18(1), 21–36. https://revistas.udistrital.edu.co/index.php/tekhne/article/view/19258

elproductor.com. (2024). Cría de codornices, una actividad rentable y económica | Noticias Agropecuarias. https://elproductor.com/2017/06/cria-de-codornices-una-actividad-rentable-y-economica/

Elsitioavicola.com. (2023). Manejo de codornices - El Sitio Avicola. https://www.elsitioavicola.com/articles/1972/manejo-de-codornices/

elsitioavicola.com. (2024). Valor nutricional y beneficios de los huevos de codorniz - El Sitio Avicola. https://www.elsitioavicola.com/poultrynews/34131/valor-nutricional-y-beneficios-de-los-huevos-de-codorniz/

García-Sucerquia, J., & Palacio-Gómez, C. (2011). Control de Temperatura Utilizando Lógica Difusa. https://api.semanticscholar.org/CorpusID:190047822

Liu, D., & Zhang, H. (2003). Fuzzy control: K. M. Passino and S. Yurkovich; 1998 Addison Wesley Longman, Inc., ISBN 0-201-18074-X. Autom., 39, 1115–1116.

https://api.semanticscholar.org/CorpusID:261785854

Lógica Difusa o Fuzzy Logic: Qué es y cómo funciona + Ejemplos. (n.d.). Retrieved November 5, 2023, from https://www.iebschool.com/blog/fuzzy-logic-que-es-big-data/

Naylampmechatronics.com. (2024). Sensor de temperatura y humedad relativa DHT22 (AM2302). https://naylampmechatronics.com/sensores-temperatura-y-humedad/58-sensor-de-temperatura-y-humedad-relativa-dht22-am2302.html

programarfacil.com. (2024). ESP32 Wifi + Bluetooth en un solo lugar. https://programarfacil.com/esp8266/esp32/

8 Anexos

8.1 Código de programación de ESP32

```
#include <Adafruit_Sensor.h>
#include <WiFi.h>
#include <ESP32Time.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>
#include <DHT_U.h>
#include <PID_FUZZY_v1.h>
#include <UbidotsEsp32Mqtt.h>
#include <Servo.h>
#include "pitches.h"

#define RES_HEATER 12
#define RES_VAPER 13

#define DHTTYPE DHT22
#define DHT_1_PIN 4
#define DHT_2_PIN 5

#define SERVO_PIN 16

#define PUMP_PIN 17
#define COOLER_PIN 18
#define BUZZER_PIN 19

#define LEVEL_SW 2
#define DOOR_PIN 33

#define BTN_OK_PIN 25

#define LDR_PIN 32
```

```
const char *UBIDOTS_TOKEN = "BBUS-8R11hyZjjKkALRbhecFs7KHkps2LJA";
const char *DEVICE_LABEL = "INCUBADORA_CODORNIZ";
const char *VARIABLE_5_LABEL = "TEMP_AVRG";
const char *VARIABLE_6_LABEL = "HUMD_AVRG";
const char *VARIABLE_7_LABEL = "DAYS";
const char *VARIABLE_8_LABEL = "LIGHT";
const char *VARIABLE_9_LABEL = "ANGLE";

const u32_t PUBLISH_FREQUENCY = 300000;
unsigned long timer_publish;

Ubidots ubidots(UBIDOTS_TOKEN);

const char* SSID = "TESIS_INCUBADORA";
const char* PASSWORD = "123456789";

// const char* SSID = "xxxxxxxx";
// const char* PASSWORD = "xxxxxxxxx";

const char* ntpServer = "pool.ntp.org";
const long gmtoffset_sec = -5*3600;
const int daylightoffset_sec = 0;

const uint32_t NVM_Offset = 0x290000;
uint8_t FLASH_Address = 0;
uint64_t epoch_init;

ESP32Time rtc;

unsigned long current_time_serial = 0;
unsigned long last_time_serial = 0;
const long treshold_serial = 10000;
```

```
unsigned long current_time_buzzer = 0;
unsigned long last_time_buzzer = 0;
const long treshold_buzzer = 5000;

unsigned long current_time = 0;
unsigned long last_time_timer = 0;
const u32_t treshold_timer = 60000;
bool motor_state = true;

unsigned long current_time_light = 0;
unsigned long last_time_light = 0;
const u32_t treshold_light = 1000;

DHT_Unified dht_1(DHT_1_PIN, DHTTYPE);
DHT_Unified dht_2(DHT_2_PIN, DHTTYPE);

float temperature_1 = 0.0;
float temperature_2 = 0.0;
float humidity_1 = 0.0;
float humidity_2 = 0.0;
float temp_average = 0.0;
float humidity_average = 0.0;
float light = 0.0;
int angle_eggs = 0;

bool first_time_cooler = true;

double Setpoint_t, Input_t, Output_t;

double Kp_t=195.0,
      Ki_t=0.00001,
      Kd_t=5.0;

double Setpoint_h, Input_h, Output_h;
```

```
double Kp_h=190,
       Ki_h=0.00001,
       Kd_h=0.8;

const int days_of_incubation = 18;
int current_days;

PID PID_FUZZY_Temperature(&Input_t, &Output_t, &Setpoint_t, Kp_t, Ki_t, Kd_t, DIRECT);
PID PID_FUZZY_Humidity(&Input_h, &Output_h, &Setpoint_h, Kp_h, Ki_h, Kd_h, DIRECT);

LiquidCrystal_I2C lcd(0x27,20,4);
bool incubation_done = false;
long cont_reset_incubation = 0;

Servo servo_tray;

void move_motor_left(){
  for(int i = 160;i>=112;i--){
     servo_tray.write(i);
     delay(100);
  }
  angle_eggs = -35;
  ubidots.reconnect();
  ubidots.add(VARIABLE_9_LABEL, angle_eggs);
  ubidots.publish(DEVICE_LABEL);
}

void move_motor_right(){
  for(int j = 112;j<=160;j++){
     servo_tray.write(j);
     delay(100);
  }
  angle_eggs = 35;
```

```
  ubidots.reconnect();
  ubidots.add(VARIABLE_9_LABEL, angle_eggs);
  ubidots.publish(DEVICE_LABEL);
}

template<typename T>
void FlashWrite(uint32_t address, const T& value) {
  ESP.flashEraseSector((NVM_Offset+address)/4096);
  ESP.flashWrite(NVM_Offset+address, (uint32_t*)&value, sizeof(value));
}

template<typename T>
void FlashRead(uint32_t address, T& value) {
  ESP.flashRead(NVM_Offset+address, (uint32_t*)&value, sizeof(value));
}

void setup() {
  Serial.begin(115200);
  pinMode(COOLER_PIN,OUTPUT);
  pinMode(PUMP_PIN,OUTPUT);
  pinMode(BUZZER_PIN,OUTPUT);
  pinMode(RES_HEATER,OUTPUT);
  pinMode(RES_VAPER,OUTPUT);

  pinMode(DOOR_PIN,INPUT_PULLUP);
  pinMode(LEVEL_SW,INPUT_PULLUP);
  pinMode(BTN_OK_PIN,INPUT_PULLUP);

  dht_1.begin();
  dht_2.begin();
  sensor_t sensor_1;
  sensor_t sensor_2;
  dht_1.temperature().getSensor(&sensor_1);
  dht_1.humidity().getSensor(&sensor_1);
```

```
dht_2.temperature().getSensor(&sensor_2);
dht_2.humidity().getSensor(&sensor_2);

digitalWrite(COOLER_PIN,LOW);
digitalWrite(PUMP_PIN,LOW);
digitalWrite(BUZZER_PIN,LOW);

setToneChannel(3);
ledcAttachPin(BUZZER_PIN, 3);

ledcSetup(4,500,8);
ledcAttachPin(RES_HEATER,4);
ledcWrite(4,0);

ledcSetup(5,500,8);
ledcAttachPin(RES_VAPER,5);
ledcWrite(5,0);

servo_tray.attach(SERVO_PIN, 0);

lcd.init();
lcd.backlight();
lcd.setCursor(0, 0);
lcd.print("** Incubadora IoT **");
lcd.setCursor(0, 2);
lcd.print(" Wifi Connecting");

WiFi.mode(WIFI_STA);
WiFi.begin(SSID, PASSWORD);
Serial.print("Connecting to WiFi...");
int cont = 0;
while (WiFi.status() != WL_CONNECTED) {
  lcd.print(".");
  cont++;
```

```
  if(cont >=2){
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("** Incubadora IoT **");
    lcd.setCursor(0, 2);
    lcd.print(" Wifi Connecting...");
  }
  if(cont >= 20){
    ESP.restart();
  }
  delay(1000);
}
Serial.println();

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("** Incubadora IoT **");
lcd.setCursor(0, 2);
lcd.print("   Wifi Connected!  ");
lcd.setCursor(0, 3);
lcd.print(" IP: ");
lcd.print(WiFi.localIP());

configTime(gmtoffset_sec, daylightoffset_sec, ntpServer);
struct tm time_info;
if(getLocalTime(&time_info)){
  rtc.setTimeStruct(time_info);
}else{
  ESP.restart();
}

ubidots.connectToWifi(SSID, PASSWORD);
ubidots.setup();
ubidots.reconnect();
```

```
  FlashRead<uint64_t>(FLASH_Address, epoch_init);
  if(epoch_init == 0xFFFFFFFFFFFFFFFF){
    FlashWrite<uint64_t>(FLASH_Address,rtc.getEpoch());
  }else{
    //FlashWrite<uint64_t>(FLASH_Address,1706911094);
    FlashRead<uint64_t>(FLASH_Address, epoch_init);
  }

  delay(3000);
  lcd.clear();

  PID_FUZZY_Temperature.SetMode(AUTOMATIC);
  PID_FUZZY_Temperature.SetOutputLimits(0,255);
  PID_FUZZY_Humidity.SetMode(AUTOMATIC);
  PID_FUZZY_Humidity.SetOutputLimits(0,255);

  current_days = (int)((rtc.getEpoch()-epoch_init)/86400)+1;

  if(current_days >= 8){
    Setpoint_t = 38.8;
    Setpoint_h = 55.0;
  }else{
    Setpoint_t = 38.3;
    Setpoint_h = 55.0;
  }
  servo_tray.write(130);
  for(int i = 130;i>=112;i--){
      servo_tray.write(i);
      delay(100);
  }

}
```

```
void loop() {
  current_days = (int)((rtc.getEpoch()-epoch_init)/86400)+1;

  sensors_event_t event_1;
  sensors_event_t event_2;

  dht_1.temperature().getEvent(&event_1);
  dht_2.temperature().getEvent(&event_2);

  if (!isnan(event_1.temperature)) {
    temperature_1 = event_1.temperature;
  }else {
    Serial.println(F("Error reading temperature_1!"));
  }
  if (!isnan(event_2.temperature)){
    temperature_2 = event_2.temperature;
  }else {
    Serial.println(F("Error reading temperature_2!"));
  }

  dht_1.humidity().getEvent(&event_1);
  dht_2.humidity().getEvent(&event_2);

  if (!isnan(event_1.relative_humidity)) {
    humidity_1 = event_1.relative_humidity;
  }else {
    Serial.println(F("Error reading humidity_1!"));
  }

  if (!isnan(event_2.relative_humidity)){
    humidity_2 = event_2.relative_humidity;
  }else {
    Serial.println(F("Error reading humidity_2!"));
  }
```

```
temp_average = (temperature_1+temperature_2)/2;
humidity_average = (humidity_1+humidity_2)/2;

current_time_light = millis();
if(current_time_light-last_time_light > treshold_light){
  light = 100.0-((analogRead(LDR_PIN)/4095.0)*100.0);
last_time_light = current_time_light;
}

Input_t = temp_average;
Input_h = humidity_average;

PID_FUZZY_Temperature.Compute();
ledcWrite(4,int(Output_t));

PID_FUZZY_Humidity.Compute();
ledcWrite(5,int(Output_h));

current_time_serial = millis();
if(current_time_serial - last_time_serial > treshold_serial){
  Serial.print("SetPoint_h:");
  Serial.print(Setpoint_h);
  Serial.print(",");
  Serial.print("Input_h:");
  Serial.print(humidity_average);
  Serial.print(",");
  Serial.print("Output_h:");
  Serial.print(Output_h);
  Serial.print(",");
  Serial.print("SetPoint_t:");
  Serial.print(Setpoint_t);
  Serial.print(",");
  Serial.print("Input_t:");
```

```
  Serial.print(temp_average);
  Serial.print(",");
  Serial.print("Output_t:");
  Serial.println(Output_t);
  last_time_serial = millis();
}

if(!digitalRead(LEVEL_SW))
  digitalWrite(PUMP_PIN,HIGH);
else
  digitalWrite(PUMP_PIN,LOW);

if(current_days > days_of_incubation){
  if(!incubation_done){
    incubation_done = true;
    lcd.clear();
    lcd.setCursor(0, 2);
    lcd.print("  Incubation Done! ");
  }
  current_time_buzzer = millis();
  if(current_time_buzzer - last_time_buzzer > treshold_buzzer){
    last_time_buzzer = millis();

    int melody[] = {
      NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4
    };
    int noteDurations[] = {
      4, 8, 8, 4, 4, 4, 4, 4
    };

    for (int thisNote = 0; thisNote < 8; thisNote++) {
      int noteDuration = 1000 / noteDurations[thisNote];
      tone(BUZZER_PIN, melody[thisNote], noteDuration);
      int pauseBetweenNotes = noteDuration * 1.30;
```

```
      delay(pauseBetweenNotes);
      noTone(BUZZER_PIN);
    }
  }
}else{
  lcd.setCursor(0, 0);
  lcd.print("Light ----> ");
  lcd.print(light);
  lcd.print(" ");
  lcd.print("%");
  lcd.print(" ");
  lcd.setCursor(0, 1);
  lcd.print("Temp -----> ");
  lcd.print(temp_average);
  lcd.print(" ");
  lcd.print((char)223);
  lcd.print("C");
  lcd.setCursor(0, 2);
  lcd.print("Humd -----> ");
  lcd.print(humidity_average);
  lcd.print(" ");
  lcd.print("%");
  lcd.setCursor(0, 3);
  lcd.print("Days ----->   ");
  char buff_days[20];
  sprintf(buff_days,"%d/%d",current_days,days_of_incubation);
  lcd.print(buff_days);
}


if(first_time_cooler && temp_average > Setpoint_t-0.1){
  digitalWrite(COOLER_PIN,HIGH);
}else{
  if(first_time_cooler){
```

```
    if(current_days >= 8){
      Setpoint_t = 38.8;
      Setpoint_h = 55.0;
    }else{
      Setpoint_t = 38.3;
      Setpoint_h = 55.0;
      digitalWrite(COOLER_PIN,LOW);
    }
  }
}

if(current_days>15){
  Setpoint_h = 65.0;
}

if(humidity_average > Setpoint_h || temp_average > Setpoint_t){
  digitalWrite(COOLER_PIN,HIGH);
  if(temp_average > Setpoint_t)
    first_time_cooler = false;
}else{
  if(!first_time_cooler){
    digitalWrite(COOLER_PIN,LOW);
    if(current_days >= 8){
      Setpoint_t = 38.8;
      Setpoint_h = 55.0;
    }else{
      Setpoint_t = 38.3;
      Setpoint_h = 55.0;
    }
  }
}

if(current_days <= days_of_incubation){
  if(digitalRead(DOOR_PIN)){
```

```
    current_time_buzzer = millis();
    if(current_time_buzzer - last_time_buzzer > treshold_buzzer){
      last_time_buzzer = millis();
      int melody[] = {
        NOTE_E5, NOTE_D5, NOTE_FS4, NOTE_GS4,
        NOTE_CS5, NOTE_B4, NOTE_D4, NOTE_E4,
        NOTE_B4, NOTE_A4, NOTE_CS4, NOTE_E4,
        NOTE_A4
      };

      int durations[] = {
        8, 8, 4, 4,
        8, 8, 4, 4,
        8, 8, 4, 4,
        2
      };

      int size = sizeof(durations) / sizeof(int);

      for (int note = 0; note < 1; note++) {
        int duration = 1000 / durations[note];
        tone(BUZZER_PIN, melody[note], duration);
        int pauseBetweenNotes = duration * 1.30;
        delay(pauseBetweenNotes);
        noTone(BUZZER_PIN);
      }
    }
  }else
    digitalWrite(BUZZER_PIN,LOW);
}

current_time = millis();
if(current_days <= 8 && current_time - last_time_timer > treshold_timer){
  if(motor_state){
```

```
    move_motor_right();
  }else{
    move_motor_left();
  }
  motor_state = !motor_state;
  last_time_timer = millis();
}

if(!digitalRead(BTN_OK_PIN)){
  cont_reset_incubation++;
  if(cont_reset_incubation>10){
    lcd.clear();
    lcd.setCursor(0, 1);
    lcd.print("  Reset Incubation!");
    lcd.setCursor(0, 2);
    lcd.print("     Days 1/");
    lcd.print(days_of_incubation);
    FlashWrite<uint64_t>(FLASH_Address,rtc.getEpoch());
    delay(5000);
    ESP.restart();
  }
  delay(800);
}else{
  cont_reset_incubation = 0;
}

if ((millis() - timer_publish) > PUBLISH_FREQUENCY) {
  ubidots.reconnect();
  ubidots.add(VARIABLE_5_LABEL, temp_average);
  ubidots.add(VARIABLE_6_LABEL, humidity_average);
  ubidots.add(VARIABLE_7_LABEL, current_days);
  ubidots.add(VARIABLE_8_LABEL, light);
  ubidots.publish(DEVICE_LABEL);
  timer_publish = millis();
```

```
  }
  ubidots.loop();
}
```

8.2 Librería PID_Fuzzy

```
#ifndef PID_v1_h
#define PID_v1_h
#define LIBRARY_VERSION1.1.1
class PID
{
  public:

  //Constants used in some of the functions below
  #define AUTOMATIC 1
  #define MANUAL   0
  #define DIRECT  0
  #define REVERSE  1
  #define P_ON_M 0
  #define P_ON_E 1

  //commonly used functions
**************************************************************************
    PID(double*, double*, double*,        // * constructor.  links the PID to the Input, Output, and
       double, double, double, int, int);//   Setpoint.  Initial tuning parameters are also set here.
                                 //   (overload for specifying proportional mode)

    PID(double*, double*, double*,        // * constructor.  links the PID to the Input, Output, and
       double, double, double, int);     //   Setpoint.  Initial tuning parameters are also set here

    void SetMode(int Mode);               // * sets PID to either Manual (0) or Auto (non-0)

    bool Compute();                       // * performs the PID calculation.  it should be
                                 //   called every time loop() cycles. ON/OFF and
                                 //   calculation frequency can be set using SetMode
                                 //   SetSampleTime respectively

    void SetOutputLimits(double, double); // * clamps the output to a specific range. 0-255 by
```

```
default, but
                                        //   it's likely the user will want to change this
depending on
                                        //   the application

  //available but not commonly used functions
*******************************************************
    void SetTunings(double, double,       // * While most users will set the tunings once in the
                double);                  //   constructor, this function gives the user the option
                                          //   of changing tunings during runtime for Adaptive control
    void SetTunings(double, double,       // * overload for specifying proportional mode
                double, int);

     void SetControllerDirection(int);   // * Sets the Direction, or "Action" of the controller.
DIRECT
                                          //   means the output will increase when error is positive.
REVERSE
                                          //   means the opposite.  it is very unlikely that this will be
needed
                                          //   once it is set in the constructor.
    void SetSampleTime(int);              // * sets the frequency, in Milliseconds, with which
                                          //   the PID calculation is performed.  default is 100



  //Display functions ****************************************************************
    double GetKp();                       // These functions query the pid for interal values.
    double GetKi();                       //  they were created mainly for the pid front-end,
    double GetKd();                       // where it's important to know what is actually
    int GetMode();                        //  inside the PID.
    int GetDirection();                   //

  private:
    void Initialize();
```

```
    double dispKp;              // * we'll hold on to the tuning parameters in user-entered
    double dispKi;              //   format for display purposes
    double dispKd;              //

    double kp;                  // * (P)roportional Tuning Parameter
    double ki;                  // * (I)ntegral Tuning Parameter
    double kd;                  // * (D)erivative Tuning Parameter

    int controllerDirection;
    int pOn;

    double *myInput;            // * Pointers to the Input, Output, and Setpoint variables
    double *myOutput;           //   This creates a hard link between the variables and the
    double *mySetpoint;         //   PID, freeing the user from having to constantly tell us
                                //   what these values are.  with pointers we'll just know.

    unsigned long lastTime;
    double outputSum, lastInput;

    unsigned long SampleTime;
    double outMin, outMax;
    bool inAuto, pOnE;
};
#endif
```