industriales
etsii

Escuela Técnica Superior de Ingeniería Industrial

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Escuela Técnica Superior de Ingeniería Industrial

DISEÑO Y DESARROLLO DE UNA CÁMARA DE CULTIVO DE BAJO COSTE PARA ESTUDIAR EL CRECIMIENTO DE PLANTAS DE FORMA CONTROLADA

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Autor: **Silvia García Bebia**

Director: Juan Antonio López Riquelme

Codirector: Alejandro Pérez Pastor

Cartagena, 12 de abril de 2020

Universidad Politécnica de Cartagena

Quiero agradecer a Don Juan Antonio López Riquelme, tutor de este proyecto,

por ofrecerme la oportunidad de realizar este trabajo así como su colaboración

y ayuda en el mismo.

Mostrar mi agradecimiento a Víctor,

por su inestimable ayuda e implicación en este proyecto.

Por supuesto agradecer a Francisco,

por su incondicional apoyo que me ha dado fuerzas cuando más las necesitaba. Pilar fundamental en mi vida.

Agradecer a aquellas personas que me han acompañado y apoyado en este viaje.

Y por último, hacer especial mención a mis padres,

los cuales son los pilares fundamentales de mi vida.

Son mis guías en el camino y gracias a ellos he llegado hasta donde estoy ahora.

Índice de contenido

Índice de ilustraciones

Índice de tablas

Capítulo 1

INTRODUCCIÓN

1.1. Introducción

Durante los últimos 20 años hemos podido observar la evolución de las cámaras de cultivo en el ámbito de la agricultura así como su utilidad. Una cámara de cultivo consiste en una unidad con un mecanismo capaz de controlar la intensidad de luz, la temperatura y la humedad por períodos programables. Esto conlleva, lógicamente, a que gane una gran importancia en el campo de la industria ya que es aplicable en procesos de producción y estandarización de cultivos vegetales, investigar patologías de plantas, desarrollo de controles de calidad, estudios de crecimiento bajo una serie de condiciones controladas e incluso en desarrollos e investigaciones en el ámbito de la biotecnología vegetal.

El uso de estas cámaras no es exclusivo para el ámbito de la industria si no que también a título de usuario se puede adquirir, de una empresa encargada de su fabricación y desarrollo, para realizar nuestros propios cultivos vegetales o de germinación de plantas. Además, su uso también se ha expandido a campos académicos en entidades docentes e investigadoras donde realizan estudios y aplicaciones y donde pueden necesarias.

Ahora bien, ¿económicamente son adquiribles estas cámaras de cultivo para cualquier persona o entidad?

Por desgracia, y como hemos podido comprobar, hoy en día estas cámaras siguen teniendo un coste muy alto y por lo tanto no accesibles para todo el mundo, además de que una gran variedad de empresas no revelan demasiada información sobre sus cámaras de cultivo.

1.2. Objetivos

Dado el elevado coste de las versiones comerciales, el objetivo principal de este proyecto es el diseño y desarrollo de una cámara de cultivo de bajo coste para estudiar el crecimiento de las plantas de forma controlada. En concreto, será necesario diseñar tanto el habitáculo de la cámara, así como todos los componentes hardware y software para su correcto funcionamiento. Esto implica que dentro del objetivo principal tendremos una serie de objetivos secundarios para cumplirlo:

- Diseño y desarrollo del habitáculo de la cámara de cultivo utilizando componentes de bajo coste.
- Diseño, integración y desarrollo de todos los componentes hardware necesarios.
- Diseño y desarrollo de los componentes software de control de la cámara, así como la interfaz de la misma.

1.3. Desarrollo de la Memoria

- **Capítulo 1: Introducción.**

  En este capítulo se realizará una breve presentación de la idea del proyecto, los objetivos de este y la estructura por la que estará formada la memoria.

- **Capitulo 2: Estado del Arte.**

  En esta sección se abordará el ámbito de las cámaras de cultivo en la actualidad, así como sus aplicaciones y por lo tanto las prestaciones básicas que debe cumplir. También se analizarán las diferentes alternativas para los componentes hardware como para el sistema software necesarios para este proyecto.

- **Capítulo 3: Descripción del sistema**

Se tratará la selección de dispositivos para el sistema software de la cámara y se desarrollará posteriormente los elementos y protocolos a mayor nivel de detalle.

- **Capítulo 4: Hardware.**

Se detallará la selección de dispositivos físicos que compondrán el sistema de la cámara de cultivo, sus características principales y se desarrollará la implementación de dichos componentes.

- **Capítulo 5: Software**

Se describirán con detalle las características del entorno que se utilizará para el desarrollo del software del sistema así como el software que se desarrollará para este caso de estudio.

- **Capítulo 6: Conclusiones y trabajos futuros.**

Se expondrán las principales conclusiones obtenidas tras la elaboración de este proyecto y los posibles trabajos futuros a realizar.

- **Capítulo 7: Referencias bibliográficas.**

Se realizará un listado de todas las fuentes consultadas para el desarrollo de este proyecto así como de la información expuesta en el mismo.

- **Anexo I: Instalación arduino IDE para ESP32.**

Se desarrollará una breve explicación de cómo se implementará el entorno informático utilizado para el desarrollo y la programación del software del sistema.

Capítulo 2

ESTADO DEL ARTE

2.1. Introducción

Este proyecto consiste en el diseño y desarrollo de una cámara de cultivo de bajo coste para el crecimiento de las plantas en un entorno totalmente controlado y funcional.

Para cumplir este objetivo será necesario, como se ha comentado previamente, diseñar el habitáculo de la cámara, así como los componentes hardware y software que serán necesarios para el correcto funcionamiento.

En este capítulo se abordará el estado actual de las cámaras de cultivo, tanto comerciales como no comerciales y se presentarán alternativas para el desarrollo de la misma. Estudiaremos y analizaremos los diversos materiales disponibles y componentes para la parte de hardware, así como los dispositivos y lenguajes de programación más adecuados para integrarlos al caso de estudio.

2.2. Cámaras de cultivo comerciales

Una cámara de cultivo es un equipo diseñado para recrear condiciones controladas de intensidad de luz, temperatura y humedad en el interior en períodos programables.

Para tomar una primera idea del funcionamiento y composición de las cámaras de cultivo se estudia en un inicio las cámaras comerciales ya existentes de la firma Binder [Binder]. Esto nos permite adquirir una referencia sobre funcionamiento, dispositivos de los que dispone y una mayor exactitud en los precios actuales de mercado. Con respecto a los precios, ha sido complicado saber cómo está el mercado, ya que varias empresas que fabrican y venden estas cámaras no hacen públicos sus precios.

Ilustración 2-1: Cámara comercial de Binder

Como se puede observar en la *Ilustración 2-1,* las cámaras de cultivo tienen esta estructura, al margen del tamaño, ya que las hay mucho más grandes pero todas están compuestas por los mismos elementos básicos (iluminación, temperatura, humedad, etc.), así como la morfología. Los precios varían pero suelen ir, aproximadamente, de unos 12.000 € hasta unos 23.000 €, aunque se sospecha que existen de un precio aun superior al comentado.

	Código de producto	Descripción	Precio	Stock	Cantidad	
+	15661197	BINDER™ Serie KBW Cámaras de crecimiento con luz	12604.00€	Fecha de envío estimada del proveedor 27-09-2019		Añadir a la cesta
+	15621207	BINDER™ Serie KBW Cámaras de crecimiento con luz	13924.00€	Fecha de envío estimada del proveedor 27-09-2019		Añadir a la cesta
+	15671207	BINDER™ Serie KBW Cámaras de crecimiento con luz	19446.00€	Fecha de envío estimada del proveedor 27-09-2019		Añadir a la cesta
+	15691207	BINDER™ Serie KBWF Cámaras de crecimiento con luz y humedad	19205.00€	Fecha de envío estimada del proveedor 27-09-2019		Añadir a la cesta
+	15611217	BINDER™ Serie KBWF Cámaras de crecimiento con luz y humedad	23049.00€	Fecha de envío estimada del proveedor 27-09-2019		Añadir a la cesta

Ilustración 2-2: Lista de precios de cámaras comerciales de Binder

2.3. Cámaras de cultivo DIY

Una alternativa a las cámaras comerciales que comienzan a aparecer son las cámaras DIY ("*Do It Yourself*"). Son cámaras realizadas de forma casera con materiales corrientes como el metacrilato, madera de contrachapado, plástico, etc.

En el ámbito de control del clima se utilizan también componentes corrientes, como por ejemplo un ventilador de un ordenador portátil para la refrigeración, calefactores eléctricos, etc.

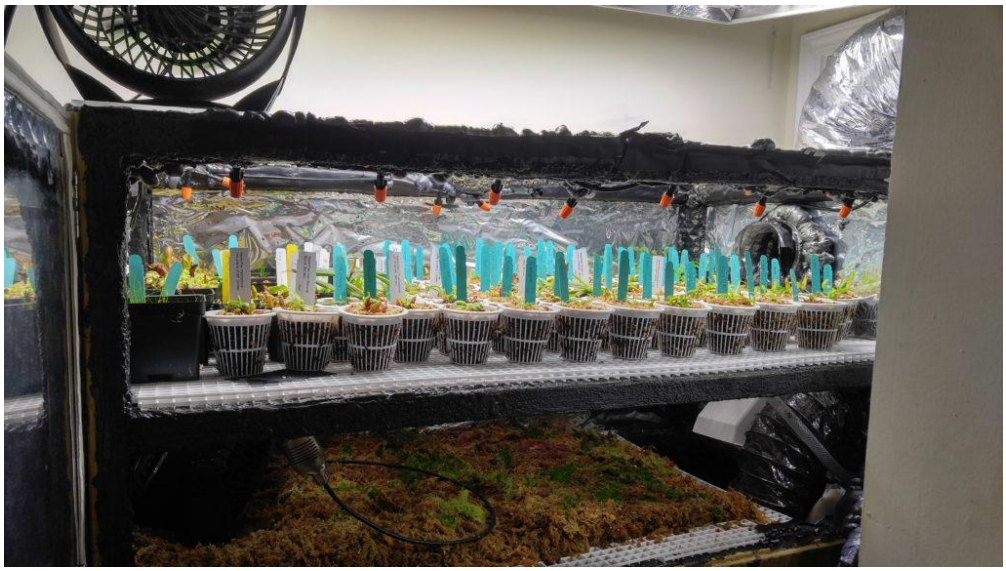

Ilustración 2-3: Cámara DIY de EdenCPs [EdenCPs]

Ilustración 2-4: Cámara DIY de un usuario de Instructables.com [Instructables]*:*

2.4. Elementos principales

A continuación se describirá la arquitectura hardware y software de la cámara a diseñar.

2.4.1. Estructura

2.4.1.1. Habitáculo

El habitáculo puede ser fabricado por diversos materiales. Ahora bien, habrá que elegir el más óptimo en relación calidad-precio que pueda satisfacer las necesidades del proyecto.

Entre la variedad de materiales se tomaron en cuenta: el metacrilato, madera de contrachapado y formica.

- Metacrilato:

  El metacrilato es un termoplástico transparente que se caracteriza por su dureza (similar al aluminio), resistencia al impacto (hasta 20 veces la del vidrio), resistencia a la intemperie y a los rayos uva (hasta 10 años), es más ligero que el cristal y no se puede doblar.

  El metacrilato es una alternativa óptima al vidrio. Es por ello que se lo conoce también como "vidrio acrílico" y es uno de los materiales plásticos de mayor consumo. Pero, a diferencia del vidrio, es irrompible, muy flexible y más transparente. Cuanto mayor sea el grosor de la lámina de metacrilato menos frágil y más duradero será.

  Por el contrario, el metacrilato se araña con cierta facilidad (por lo que no es adecuado para usos que implican desgaste mecánico) y atrae el polvo. Y, con el tiempo, tiende a amarillearse. No obstante, las características positivas del metacrilato lo convierten en un material adecuado para muchos usos: desde la arquitectura hasta la biomedicina, pasando por la ingeniería.

  Las aplicaciones del PMMA (polimetilmetacrilato) se extienden también a otros productos como urnas, tapas, trofeos, vitrinas, decoración y objetos, cajas para alimentación, señalización, expositores, etc.

Ilustración 2-5: Placas de metacrilato transparentes

- Madera de contrachapado:

De todos los materiales derivados de la madera, el contrachapado es sin duda el más conocido. El contrachapado, es un tablero elaborado con finas chapas de madera pegadas con las fibras transversalmente una sobre la otra con resinas sintéticas mediante fuerte presión y calor. De esta forma se obtiene un gran material, perfectamente plano y rigurosamente calibrado.

Este material es muy utilizado en la fabricación de muebles, suelos, armarios, estructuras, techos, etc. Este uso extendido se debe a su gran resistencia, su capacidad de uso y variedad de acabados y espesores.

Por el contrario, se puede expandir al ser expuesto a altas temperaturas o a la humedad y puede absorber fácilmente el agua o vapor de agua por lo que puede sufrir deformidades (aun así, este problema se puede prevenir mediante selladores y barnices que protejan la madera de la humedad).

Ilustración 2-6: Tablas de madera de contrachapado

- Formica:

El laminado plástico Formica es un revestimiento con características de adaptabilidad, resistencia y durabilidad que tiene una gran variedad de aplicaciones. A menudo se confunde Formica con melamina, la Formica es una marca de laminado a alta presión (HPL) y la melamina el nombre genérico de los laminados a baja presión (LPL). Es económica.

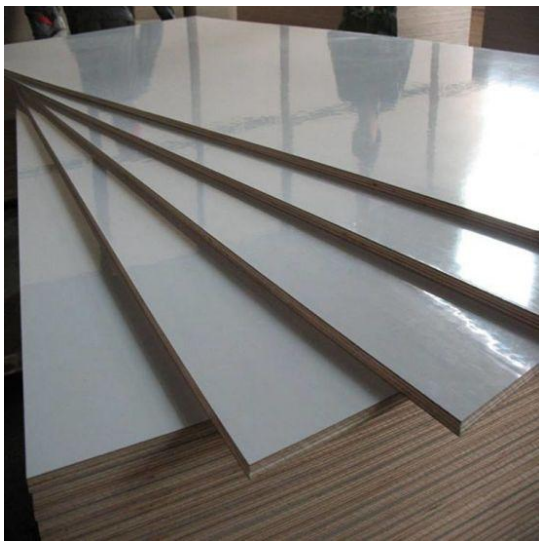

Ilustración 2-7: Láminas de formica

2.4.1.2. Sellado

Un aspecto muy importante es que el habitáculo tenga un buen sellado en las juntas para evitar fugas y fallos de rendimiento.

Existen una variedad de selladores:

- Sellador acrílico para madera.
- Cinta de sellado de silicona.
- Tiras de sellado de goma espuma.
- Masillas.

2.4.1.3. Aislamiento

Otro aspecto muy importante es que el habitáculo esté bien aislado. Un buen aislante térmico genera:

- Confort térmico: protege de los días más fríos y de los días más calurosos.
- Confort acústico: dada sus prestaciones técnicas también es capaz de proteger del ruido externo.
- Protección contra las humedades: prevención contra las humedades por condensación.
- Ahorro energético y económico: al haber más protección contra condiciones climáticas, se reduce la demanda energética y por lo tanto el costo energético.
- Duración: prestaciones ilimitadas hasta la vida útil del habitáculo.

Por lo tanto, los aislantes que se utilizan para lograr el máximo ahorro de energía y con las mejores prestaciones técnicas en relación calidad/precio son los que van a cobrar más protagonismo en nuestro proyecto.

Poliestireno expandido

Un material muy usado hoy en día y con buenos resultados es el poliestireno expandido (EPS). El poliestireno expandido es un material de origen sintético el cual se conoce como corcho blanco o porespan y viene en planchas de distintos espesores según se requiera mayor o menor aislamiento térmico, densidades que van desde 10 hasta 25 kg/m^3 y con una conductividad térmica de entre 0,029 y 0,053 W/(mK). Tiene la ventaja de tener un buen comportamiento térmico a bajas densidades. Otra de sus ventajas es que se puede manipular sin necesitar medidas de protección y se utiliza como material para aligerar además de ser usado como aislante térmico. Una desventaja es que es atacable por los rayos ultravioletas.

El poliestireno expandido comparte muchas características con el extruido, su composición es aproximadamente un 95% poliestireno y un 5 % gas. Sin embargo el proceso de fabricación determina una diferencia fundamental: el extruido tiene estructura cerrada, por lo que es un aislamiento térmico que puede mojarse sin perder sus propiedades.

Ilustración 2-8: Panel de poliestireno expandido

Las principales diferencias entre el poliestireno expandido y el extruido son las siguientes:

- El poliestireno expandido es menos denso.
- Por lo tanto no puede ir machihembrado.
- Al tener estructura abierta absorbe la humedad, a diferencia del extruido.
- Tiene una resistencia mecánica menor.

Poliestireno extruido

El poliestireno extruido (XPS) es un material, como ya hemos dicho antes, muy similar al EPS pero con otras propiedades. Quizá la más importante es que puede mojarse, por lo que se instala mucho en cubiertas. Normalmente se sirve machihembrado, en planchas y con espesores típicos de 40 / 50 / 60 / 80 mm.

Su conductividad térmica se encuentra entre 0,025 y 0,040 W/mK.

La baja absorción de agua y la resistencia a los ciclos de hielo – deshielo lo hacen ideal para cubiertas en las que el aislante se coloca inmediatamente debajo de la teja. Por otra parte su gran resistencia mecánica permite que las cargas (peso de tejas, nieve, presión / succión de viento) puedan apoyar directamente sobre el aislante.

Se puede usar como aislamiento en tabiquería, con paneles que van de forjado a forjado, en cubierta (paneles de chapa), muros enterrados, techos, entre otros.

Ilustración 2-9: Panel de poliestireno extruido

Espuma de poliuretano

Otro producto que brinda excelentes resultados es la espuma de poliuretano (PUR). Es un producto cuya composición básica es petróleo y azúcar, formándose una espuma rígida ligera con más del 90% de las celdas cerradas y buen coeficiente de conductividad térmica (muy aislante), comprendido entre 0,019 y 0,040 W/(mK).

Tiene un coeficiente de conductividad muy bajo por lo que no necesita ser muy gruesa para obtener un buen coeficiente de aislamiento. Debe estar muy bien aislada de posibles incendios porque al quemarse desprende ácido cianhídrico que es tóxico para el ser humano.

Como características principales podemos mencionar su rigidez estructural, una gran adherencias sobre distintas superficies, baja o nula absorción de humedad y buena relación aislamiento / precio. Tiene la ventaja, además, de su aplicación con pistola en forma de espuma, rellenando cámaras y huecos.

Otro uso típico en paneles sándwich, compuestos de dos capas metálicas y material de aislamiento (espuma de poliuretano, lana de roca, poliestireno) entre ellas. Son modulares y ligeros.

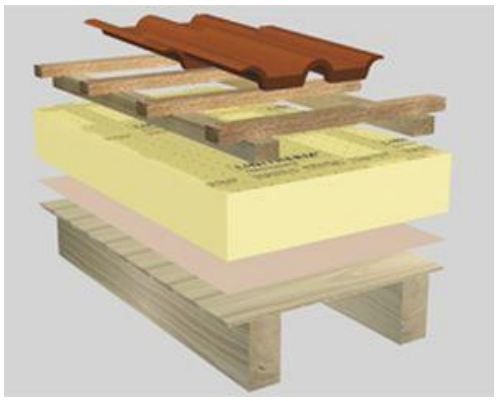

Ilustración 2-10: Espuma de poliuretano

Ilustración 2-11: Espuma de poliuretano proyectado

Espuma elastomérica

Dentro de las espumas aislantes también se encuentra la espuma elastomérica, la cual tiene un coeficiente de conductividad muy similar al poliuretano. Tiene la ventaja de que se puede instalar fácilmente pero, al igual que el corcho blanco, los rayos ultravioletas lo atacan.

Lana de roca

Los paneles de lana de roca están compuestos casi en su totalidad de roca de origen volcánico con un pequeño porcentaje de ligante orgánico. Se obtiene fundiendo la roca a altas temperaturas, sometiéndole a movimientos y aplicando aglomerantes y aceites impermeables, transformándose después en paneles, mantas, etc.

A diferencia de los poliestirenos EPS / XPS y el poliuretano, las lanas minerales no son inflamables, pero para su colocación hay que protegerse los ojos, la piel y el sistema respiratorio. La conductividad térmica de las lanas minerales (de roca y vidrio) se encuentra entre 0,03 y 0,05 W/(mK).

La lana de roca gracias a su disposición multidireccional de fibras tiene también una buena capacidad como aislante acústico.

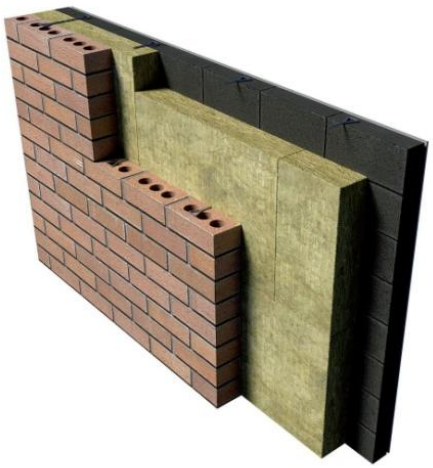

Ilustración 2-12: Aislamiento térmico con lana de roca

Lana de vidrio

No hay que olvidar La lana de vidrio o fibra de vidrio, producto de origen natural, mineral e inorgánico (arena de sílice, carbonato de calcio y de magnesio), compuesto por filamentos de vidrio aglutinados mediante resina ignífuga. Se obtiene por un proceso similar a la lana de roca y presenta buena resistencia a la humedad.

Al igual que la lana de roca se sirve en forma de mantas y paneles, siendo un aislamiento térmico ignífugo.

Vidrio expandido

Por último, otro gran aislante es el vidrio expandido. Este material, además de sus cualidades térmicas, cumple muy bien el rol de barrera de vapor. Esto lo hace apto para evitar puentes térmicos como uniones entre muros y pilares, carpintería y mampostería por ejemplo.

2.4.2. Iluminación

Encontrar instalada iluminación LED en un cultivo interior empieza a no ser sorprendente. Con el paso de los años, esta técnica ha ganado mucho por sus enormes ventajas para cualquier tipo de cultivo y su versatilidad. El gran ahorro energético que generan la iluminación LED es una de las razones principales de este auge: dependiendo del tipo de planta, hay cultivos que pueden necesitar hasta 18 horas ininterrumpidas de luz. Con el uso específico de lámparas LED para cultivo interior el gasto puede llegar a ser 25 veces mejor que respecto a determinadas luces tradicionales.

La luz que diariamente reciben las plantas es fundamental para activar la fotosíntesis. Si una planta no recibe la luz necesaria o no crecerá o lo hará en malas condiciones

Además, gracias a la utilización de las denominadas “lámparas Grow” se puede manipular fácilmente el espectro de luz (sobre todo debido a los diferentes colores que se pueden usar) que reciben las plantas de un cultivo de interior. Con la sustitución de los viejos sistemas por otros más actuales, se pueden realizar instalaciones en las que se emiten espectros separados y necesarios para el crecimiento y floración de las plantas.

La temperatura también se erige como un valor diferencial: los LEDs no emiten tanto calor como las bombillas tradicionales, por lo que se pueden colocar más cerca de las plantas sin miedo a dañarlas.

Gracias a los focos LED para cultivo interior las plantas pueden absorber mucha más cantidad de luz que con las lámparas halógenas. Y por su ya mencionada versatilidad se pueden instalar tanto en invernaderos, cámaras de crecimiento controlado o acuarios.

Ilustración 2-13: Iluminación LED para el crecimiento de plantas

A continuación, estos son algunos LED que se pueden utilizar en función de cada necesita concreta y que, por supuesto, se pueden instalar combinados para que cada uno ejecute su función:

- **LEDs rojos**: no solo aportan los rayos infrarrojos que cualquier planta necesita para florecer, sino que además es un perfecto repelente natural de insectos.
- **LEDs blancos**: a falta de luz natural son la mejor alternativa para lograr algo parecido a ella.
- **LEDs azules**: imprescindibles para aportar los rayos UV que requieren las plantas para crecer correctamente.

Sabiendo que los LEDs encarnan una poderosa herramienta para sacar todo el provecho a un cultivo de interior hay algunas características que también vienen bien comentar:

- Cuando las plantas están empezando a crecer es cuando más luz necesitan. Se debe seleccionar correctamente la lámpara LED necesaria y adaptar la separación entre la fuente de luz y las plantas, aunque sabiendo que nunca se quemarán.
- Un buen temporizador es una ayuda adicional perfecta para controlar los flujos de luz que necesita cada planta diariamente.

Teniendo en cuenta lo comentado, se procederá a la elección directa de LEDs para la iluminación de las plantas o semillas que se tengan en el interior del habitáculo. Encontramos varios formatos de entre los que se estudian los siguientes:

- Tubos LED: El tubo de LEDs para cultivo está diseñado especialmente para uso durante todo el ciclo de crecimiento de las plantas, desde su germinación hasta su maduración. Fabricado con cuerpo de aluminio que ofrece mayor rigidez y mejor disipación de calor por lo que lo hacen apropiado para encendidos continuos.

- Cinta LED: La iluminación mediante tiras de LEDs especiales para el crecimiento, ofrece múltiples ventajas además del ahorro energético, ya que al manipular el espectro de luz que reciben las plantas que están creciendo, aumenta exponencialmente la producción de los cultivos sin afectar la calidad respecto a otras formas de iluminación tradicional, consumiendo 5 veces menos energía y evitando el exceso de calor dañino a las plantas. Además, se puede cortar cada 3 LEDs en función de la distribución que se tenga por lo que permite adaptarlo.

2.4.3. Control de temperatura

Se utilizará un módulo de temperatura que permitirá medir la temperatura del interior y tomar los datos para programar el sistema de refrigeración o calefacción con el que se mantendrá la temperatura deseada en el interior.

2.4.3.1. Sistema de refrigeración

- Refrigeración pasiva por aire:

Es el método más antiguo y común para enfriar no sólo componentes electrónicos sino cualquier cosa. Los componentes para ello suelen ser disipadores que consisten en cientos de aletas delgadas. Mientras más aletas, más disipación. Mientras más delgadas, mejor.

Las principales ventajas de la disipación pasiva son su inherente simplicidad, su durabilidad (pues carece de piezas móviles) y su bajo costo. Además de lo anterior, no producen ruido. La mayor desventaja de la disipación pasiva es su habilidad limitada para dispersar grandes cantidades de calor rápidamente.

- Refrigeración activa por aire:

La refrigeración activa por aire es, en palabras sencillas, tomar un sistema pasivo y adicionar un elemento que acelere el flujo de aire. Este elemento es usualmente un ventilador aunque se han visto variantes en las que se utiliza una especie de turbina.

Aunque la refrigeración activa por aire no es mucho más cara que la pasiva, la solución tiene desventajas significativas. Por ejemplo, al tener partes móviles es susceptible de averiarse, pudiendo ocasionar daños. En segundo lugar, aunque este aspecto ha mejorado mucho todos los ventiladores hacen ruido. Algunos son más silenciosos que otros, pero siempre serán más ruidosos que los cero decibelios que produce una solución pasiva.

- Refrigeración líquida:

Un método más complejo y menos común es la refrigeración por agua. El agua tiene un calor específico más alto y una mejor conductividad térmica que el aire, gracias a lo cual puede transferir calor más eficientemente y a mayores distancias que el gas. Bombeando agua es posible remover grandes cantidades de calor de éste en poco tiempo, para luego ser disipado por un radiador ubicado en algún lugar dentro (o fuera).

La principal ventaja de la refrigeración líquida es su habilidad para enfriar pero tiene un coste relativamente caro, es compleja e incluso peligrosa en manos sin experiencia. Aunque usualmente menos ruidosos que los basados en refrigeración por aire, los sistemas de refrigeración por agua tienen partes móviles y en consecuencia se sabe eventualmente pueden sufrir problemas.

- Refrigeración termoeléctrica:

En 1834 un francés llamado Jean Peltier, descubrió que aplicando una diferencia eléctrica entre 2 metales o semiconductores (de tipo p y n) unidos entre sí, se generaba una diferencia de temperaturas entre las uniones de estos.

El concepto rudimentario de Peltier fue paulatinamente perfeccionado para que fuera un solo bloque (conocido posteriormente como *célula Peltier*) con las uniones semiconductoras conectadas por pistas de cobre y dispuestas de tal manera , que transportara el calor desde una de sus caras hacia la otra, haciendo del mecanismo una “bomba de calor”, ya que es capaz de extraer el calor de una determinada superficie y llevarlo hacia su otra cara para disiparlo.

Son bastante versátiles, basta con invertir la polaridad para invertir el efecto (cambiar el lado que se calienta por el frío y viceversa). La potencia con que enfría es fácilmente modificable dependiendo del voltaje que se le aplique y es bastante amable con el medio ambiente, ya que no necesita de gases nocivos como los usados en los refrigeradores industriales para realizar su labor.

Por lo tanto, la tecnología Peltier es especialmente rentable con temperaturas cercanas a la temperatura ambiente y permite ahorrar energía, ya que únicamente se utiliza energía cuando se realizan los procesos de refrigeración o calentamiento, a diferencia de los sistemas con compresor. Al mismo tiempo, es posible ajustar la función de refrigeración o calentamiento con gran precisión.

Tras mencionar su versatilidad y como se puede invertir la polaridad de las células Peltier, es necesario destacar la funcionalidad que tienen como bomba de calor electrónica donde la energía térmica se transportará del lado frío al lado caliente. Es decir, en función de la dirección del flujo de corriente, se pueden utilizar para refrigerar o para calentar.

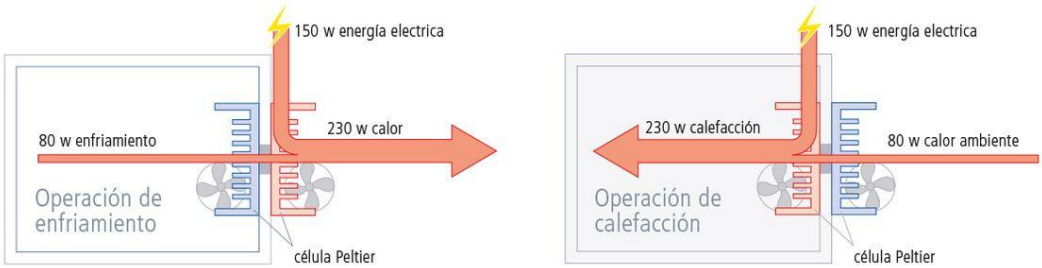

Ilustración 2-14: Diagrama de funcionamiento de un kit de Peltier

Estos kits existen de muchos tamaños y tipos:

- Termoeléctrica Peltier Refrigeración Cooler Kit Semiconductor enfriador.

Ilustración 2-15: Kit Peltier de refrigeración con dos ventiladores: **[Amazón]**

- Kit termoeléctrico de enfriamiento Peltier.

Ilustración 2-16: Kit Peltier de refrigeración con tres ventiladores: [Amazón]

- Kit de Bricolaje de Refrigeración de Semiconductores de 12 V Sistema de Deshumidificación de Enfriamiento por Aire Peltier Termoeléctrico.

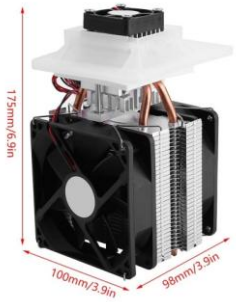

Ilustración 2-17: Kit Peltier de refrigeración para deshumidificar: [Amazón]

2.4.3.2. Sistema de calefacción

- Calefactor eléctrico: se recomiendan para zonas pequeñas, ya que su alcance es menor que el de otros dispositivos de calefacción.
  - **Ventajas**
    - Rapidez de calefacción gracias al ventilador que llevan incorporado.
    - El calor se propaga rápidamente y se focaliza en el punto que nosotros deseamos.
    - Fáciles de usar y manejar gracias a su tamaño reducido.

- **Inconvenientes**
  - Están indicados como complementos a otras fuentes de calor, ya que si se usan como fuente principal, elevan mucho el coste energético.
  - Consumen oxígeno; por lo tanto, en estancia cerradas, su uso por mucho tiempo puede producir una sensación de ambiente cargado.

2.4.4. Control de humedad relativa

Se instalará un módulo de humedad relativa que medirá la humedad ambiente, así como tomar los datos necesarios y programar la humidificación o deshumidificación del interior de la cámara para mantener la humedad deseada.

Ilustración 2-18: Modulo DHTP22 sensor de temperatura y humedad

Para humidificar se utilizará humidificadores para habitáculos cerrados. Dispositivo que se encarga de aportar la humedad necesaria al ambiente si además se tiene un calefactor que genera un calor seco que reseca el ambiente. Los hay de diversos tamaños y características que se analizarán más adelante.

Ilustración 2-19: Humidificador para entornos cerrados

Para deshumidificar se estudian dos posibilidades:

- **Deshumidificar mediante ventilación natural de la cámara.**

Por ventilación natural se entiende aquellos procesos de entrada de aire seco y limpio del exterior sin la utilización de máquinas o equipos.

Se realiza a través de los orificios existentes en los paramentos exteriores de las construcciones y mediante las operaciones manuales de apertura de puertas y ventanas en ciertos momentos.

- **Deshumidificar mediante ventilación mecánica de la cámara.**

Instalación de extractores mecánicos o mediante una serie de ventiladores distribuidos estratégicamente para provocar una circulación de aire y extraerla o más bien conducirla hacia el exterior del habitáculo. Esta circulación de aire arrastrará los excesos de humedad e incluso de temperatura que existan en el interior.

- **Usar un deshumidificador para habitáculos cerrados.**

Uso de dispositivos mecánicos que permiten la captura del vapor de agua presente en el aire y lo condensan en su interior de donde deberá ser eliminado posteriormente. Se puede asociar su puesta en marcha con detectores de humedad y programadores automáticos de forma que dota de cierta automatización al conjunto. Puede ser una alternativa para pequeñas construcciones.

2.4.5. Interfaz

Para la interfaz se estudiarán dos opciones:

- Panel de control en el exterior de la cámara:

Panel donde se visualice el estado del interior de la cámara (valor de temperatura, humedad relativa, iluminación) y permita interactuar y programar la cámara.

- Uso de aplicación móvil mediante Bluetooth:

Diseño de una aplicación móvil conectada vía Bluetooth con la placa de control de la cámara que permitirá consultar en tiempo real el estado y manipular y ajustar parámetros del interior de la cámara.

2.4.6. Sistema de procesamiento

Para llevar a cabo este proyecto y la automatización y control de los procesos se requiere de un sistema de procesamiento. Se ha elegido trabajar con el entorno de Arduino debido a las facilidades que otorga, así como su sencillez a la hora de trabajar y programar.

Como se va a utilizar el entorno de Arduino, las alternativas para la elección del sistema de procesamiento girarán en torno a ello.

ESP8266

El ESP8266 es un chip Wi-Fi de bajo coste con pila TCP/IP completa y capacidad de MCU (Micro Controller Unit) producida por el fabricante chino Espressif Systems [Espressif]. La primera placa de desarrollo con este chip llegó en agosto de 2014 con el módulo ESP-01. Este pequeño módulo permite a los microcontroladores conectarse a una red Wi-Fi y realizar conexiones TCP/IP sencillas utilizando comandos de tipo Hayes. Sin embargo, en ese momento casi no había documentación en inglés sobre el chip y los comandos que aceptaba. Su bajo coste y el hecho de que requería de pocos componentes externos, atrajo a muchos hackers para explorar el módulo, el chip y el software en él, así como para traducir la documentación del mismo a otros idiomas.

El ESP8285 es un ESP8266 con 1 MB de flash incorporado, lo que permite dispositivos de un solo chip trabajar con redes Wi-Fi. Muchos encapsulados del ESP8266 vienen con 1 MB de flash.

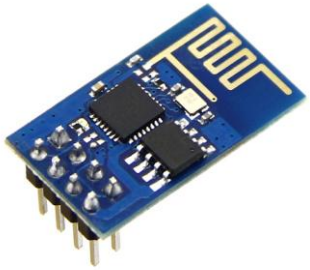

Ilustración 2-20: Tarjeta ESP8266

Con respecto a la tensión de alimentación, el ESP8266 es un módulo que va alimentado a 3,3 V.

ESP32

El ESP32 es una serie de microcontroladores de bajo coste, con un sistema de bajo consumo de energía, Wi-Fi integrado y Bluetooth. La serie ESP32 es la sucesora de la serie ESP8266 y utiliza un microprocesador de doble núcleo.

Aunque en próximos capítulos se profundizará más en este microprocesador, si se puede destacar que es un dispositivo muy versátil, que se puede programar en el entorno Arduino de manera sencilla y además lleva incorporado conexión WiFi y Bluetooth. Este último punto se tendrá muy en cuenta, ya que este protocolo de comunicación es esencial si se opta por una determinada interfaz de usuario.

STM32

Como se desarrollará en los próximos capítulos, STM32 es una familia de microcontroladores que en este caso se tienen en cuenta para el control de la iluminación LED y que estará comunicado mediante un bus con la tarjeta que seleccionemos como tarjeta de control del sistema.

Es una placa ARM barata y versátil, y a pesar de que no está incluida en un inicio en el entorno de Arduino, tiene una comunidad que ha desarrollado las herramientas necesarias para incluirlo, por lo que también se podrá trabajar con él a través de Arduino.

Capítulo 3

DESCRIPCIÓN DEL SISTEMA

3.1. Introducción

En el capítulo anterior se ha estudiado las diversas opciones para cada elemento del proyecto propuesto. Como se indicó en dicho capítulo, la cámara de cultivo irá automatizada y por lo tanto requerirá del uso de microcontroladores, sistemas de comunicación y programación.

Para el sistema software del caso de estudio se propone I^2C como sistema de comunicación entre los microcontroladores y hardware de control, el módulo ESP32 como hardware de control para la temperatura, humedad e iluminación de la cámara y Android como sistema operativo en el que se desarrollará la aplicación que se usará como interfaz entre la cámara de cultivo y el usuario.

Además, se ha elegido Arduino debido a su sencillez de configuración y programación. También añadir que se trata de un software libre. Se ha decidido usar Android por su facilidad y simplicidad a la hora de diseñar una aplicación. Por último, ha sido elegido el módulo ESP32 por su bajo coste y por su sencillez a la hora de programar y controlar un sistema automatizado.

3.2. Arquitectura del sistema

La estructura del proyecto se inicia desde el instante en el que, una vez puesta en marcha la cámara, los diversos sensores instalados en su interior comienzan a tomar datos del medio interno que se ha recreado hasta el comienzo del sistema de control y automatización pasando por una serie de etapas.

A continuación se resumen los diferentes componentes que forman parte de las fases que se van a ir desarrollando:

- Sensor de temperatura: dispositivo cuya función será la lectura de la temperatura en el interior del habitáculo.

- Sensor de humedad relativa: dispositivo encargado de la lectura de la humedad relativa en el interior del habitáculo.

- Iluminación LED: distribución de una serie de LEDs para cultivo de interior que aportarán la iluminación necesaria para el cultivo introducido en el interior.

- Arduino Nano: microcontrolador basado en el microcontrolador ATmega328 que se caracteriza por su pequeño tamaño y que tiene la capacidad de aplicar las instrucciones de control.

- ESP32: placa de control encargada del sistema de control de todos los elementos del proyecto. Además incorpora conexión Bluetooth 4.0 para, por ejemplo, una aplicación Android. Finalmente, el módulo ESP32 también es compatible con diferentes protocolos de comunicación.

- Aplicación Android: realiza la función de interfaz entre el usuario y la cámara y permite al usuario controlar a distancia, mediante conexión Bluetooth, cualquier característica configurable de la cámara.

3.3. Sensor de temperatura

Un sensor es un objeto capaz de detectar magnitudes físicas o químicas llamadas variables de instrumentación. Por lo tanto, el sensor de temperatura es un dispositivo que transforma los cambios de temperatura en señales eléctricas que son procesados por un equipo eléctrico o electrónico. Estas diferencias de temperatura son detectadas a partir de unos materiales metálicos unidos mediante unión fría y unión caliente. El grado de temperatura que detectan estas uniones genera una diferencia de potencial estrechamente dependiente de la naturaleza de los materiales la cual es la que el sensor transforma en señal eléctrica.

El sensor de temperatura suele estar formado por el elemento sensor, que puede ser de diferentes tipos (termistor, RTD o termopar), la vaina que lo envuelve y que está rellena de un material muy conductor de la temperatura, para que los cambios se transmitan rápidamente al elemento sensor y del cable al que se conectarán el equipo electrónico.

Termistor

Un termistor es un sensor que sirve para detectar temperatura a través de cambios de resistencia según el calor o el frío detectado. Existen 2 tipos de sensores según su coeficiente de temperatura: NTC y PTC

- **NTC** (*Negative Temperature Coefficient*): resistencia con coeficiente de temperatura negativo con respecto a la variación de su resistencia, es decir, a más temperatura su resistencia será menor.

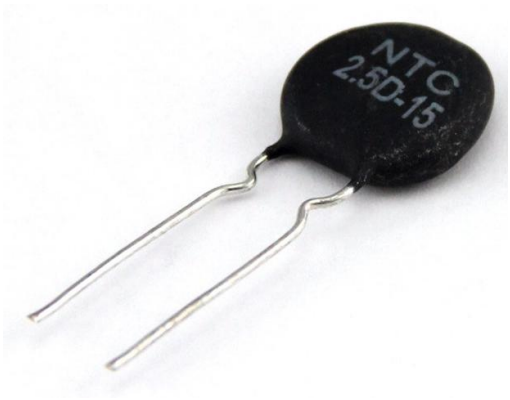

Ilustración 3-1: Termistor NTC 2.5D-15

- **PTC** (*Positive Temperature Coefficient*): resistencia cuyo coeficiente de temperatura es positivo (a más temperatura su resistencia aumentará también).

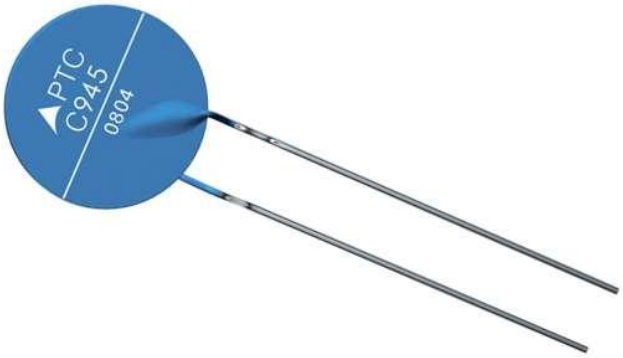

Ilustración 3-2: Termistor PTC C945

RTD

Es un sensor de temperatura compuesto por un alambre fino el cual tiene una relación temperatura-resistencia que al aumentar la temperatura incrementa su resistencia. Como su respuesta es lineal se puede saber el nivel de calor utilizando una tabla que muestra los valores resistivos que se generan en cada grado centígrado o Fahrenheit. Debido a que los componentes este tipo de sensor puede ser muy delicado por lo que generalmente se encuentran en un encapsulado de acero inoxidable como el de los termopares.

Temperatura °C	Resistencia Ω
0	100
10	103.9
20	107.79
30	111.67
40	115.54
50	119.4

Ilustración 3-3: RTD Pt 100

Según como estén diseñados se pueden encontrar cuatro tipos diferentes de RTDs:

- **Tipo bobinado**: diseñado con una cubierta de cerámica y un bobinado en el núcleo. El enrollado de la bobina puede ser circular o plano, pero siempre debe de estar acompañado de algún aislante eléctrico.
- **Tipo laminado**: fabricado por una delgada capa de platino y cubierto con una resina o vidrio que ayuda a proteger la envoltura de platino disminuyendo la deformación de los cables.
- **Tipo enroscado**: construido a partir de una bobina helicoidal de alambre de platino. Los conductores se insertan a través de un tubo de óxido de aluminio con 4 agujeros. Los orificios son rellenados con polvo cerámico, evitando cortocircuitos y posibles vibraciones durante la medición.
- **Tipo de anillo hueco**: este tipo de sensor utiliza un metal de composición abierta aumentando el fluido del contacto con masa térmica pequeña, lo que ayuda a proporcionar un tiempo de respuesta más rápido. Su área externa está totalmente recubierta con material aislante. Una desventaja de estos tipos de sensores es su elevado coste.

Termopar

Un termopar es un sensor para medir la temperatura. Este sensor se compone de dos metales o aleaciones de metales diferentes unidos en un extremo y separados por el otro. Cuando en sus uniones existe una diferencia de temperatura, se origina una fuerza electromotriz o diferencia de tensión, conocida como el efecto Seebeck (efecto termoeléctrico). Este efecto recibe su nombre de Thomas Johann Seebeck, científico que lo descubrió en el año 1821. La diferencia de tensión generada por el termopar está en función de la diferencia de temperatura entre la unión fría (parte separada) y caliente (parte unida), lo que significa que un termopar no mide la temperatura absoluta si no la temperatura diferencial entra la unión caliente y la unión fría.

Ilustración 3-4: Disposición de unión caliente con unión fría

En resumen, cuando la unión de los dos metales se calienta o se enfría, se produce una tensión que es proporcional a la temperatura.

Ilustración 3-5: Termopar

Cuando se origina dicho cambio en la temperatura de la punta de los cables, la diferencia de potencial que genera se representa en forma de una tensión eléctrica muy pequeña (del orden de mV) que muchos dispositivos no pueden leer.

Esta tensión del termopar se introduce en un controlador que forma parte del dispositivo y que interpreta la señal. Normalmente, la señal generada se amplifica para que pueda ser leída por el controlador y convertida a una lectura de temperatura en grados centígrados o Fahrenheit.

Además, dentro de la familia de los termopares existen diferentes combinaciones de metales o calibraciones para adaptarse a diferentes aplicaciones. Los tres más comunes son las calibraciones tipo J, K y T, de los cuales el termopar tipo K es el más popular debido a su amplio rango de temperaturas y bajo coste.

- **Tipo K** (cromel/alumel): tiene una amplia variedad de aplicaciones, está disponible a un bajo costo y en una variedad de sondas. El cromel es una aleación de Ni-Cr, y el alumel es una aleación de Ni-Al. Tienen un rango de temperatura de –200 °C a +1372 °C y una sensibilidad 41 µV/°C aproximadamente. Además, posee buena resistencia a la oxidación.
- **Tipo J** (hierro/constantán [aleación de Cu-Ni]): su rango de funcinoamiento es de –270/+1200 °C.

Debido a sus características se recomienda su uso en atmósferas inertes, reductoras o en vacío. Su uso continuado a 800 °C no presenta problemas. Su principal inconveniente es la rápida oxidación que sufre el hierro por encima de 550 °C y por debajo de 0 °C es necesario tomar precauciones a causa de la condensación de vapor de agua sobre el hierro.

- **Tipo T**: (cobre/constantán): son ideales para mediciones entre -200 y 260 °C. Resisten atmósferas húmedas, reductoras y oxidantes, y son aplicables en criogenia. El tipo termopar de T tiene una sensibilidad de cerca de 43 µV/°C.

Características	RTD	Termistor
Precisión	Más preciso	Menos preciso
Rango de Tª	-200 a 850 °C	-250 a 2000 °C
Coste	Más caro	Más barato
Respuesta	Más lenta	Más rápida
Tamaño	Más largo	Tan pequeño como sea posible
Alimentación	Requerida	No requerida
Estabilidad con periodos largos	Excelente	Menos satisfactoria
Salida	Ω	mV

Tabla 3-1: Comparativa RTD - Termistor

3.4. Sensor de humedad relativa

Los sensores de humedad miden el nivel de líquido o la humedad relativa en un área dada, permiten controlar la humedad del aire y la temperatura. Las magnitudes medidas por el sensor de humedad se transforman en una señal eléctrica. Estos sensores pueden ser sensores analógicos de humedad o sensores digitales de humedad.

Un sensor analógico de humedad mide la humedad del aire relativo usando un sistema basado en un condensador. El sensor está hecho de una película generalmente de vidrio o de cerámica. El material aislante que absorbe el agua está hecho de un polímero que toma y libera el agua basándose en la humedad relativa de la zona dada. Esto cambia el nivel de carga en el condensador del circuito en el cuadro eléctrico.

Un sensor digital de humedad funciona a través de dos micro-sensores que se calibran a la humedad relativa de la zona dada. Estos se convierten luego en el formato digital a través de un proceso de conversión de analógico a digital que se realiza mediante un chip situado en el mismo circuito. Un sistema basado en una máquina hecha de electrodos con polímeros es lo que constituye la capacitancia del sensor. Esto protege el sensor del panel frontal del usuario.

Según el principio físico que siguen para realizar dicha medición existen de varios tipos:

- **Mecánicos**: aprovechan los cambios de dimensiones que sufren cierto tipos de materiales en presencia de la humedad (fibras orgánicas o sintéticas, el cabello humano, etc.).
- **Basados en sales higroscópicas**: deducen el valor de la humedad en el ambiente a partir de una molécula cristalina que tiene gran afinidad con la absorción de agua.
- **Por conductividad**: la presencia de agua en un ambiente permite que a través de unas rejillas de oro circule una corriente ya que el agua es buena conductora de corriente. Según la medida de corriente que se obtiene, se deduce el valor de la humedad.
- **Capacitivos**: se basan en el cambio de la capacidad que sufre un condensador ante la presencia de humedad.
- **Infrarrojos**: estos disponen de dos fuentes infrarrojas que lo que hacen es absorber parte de la radiación que contiene el vapor de agua.
- **Resistivos**: aplican un principio de conductividad de la tierra. Es decir, cuanta más cantidad de agua hay en la muestra, más alta es la conductividad de la tierra.

Ilustración 3-6: Sensor de humedad relativa Hs1101

3.5. Iluminación LED

El LED es un claro candidato a ser el futuro del cultivo de interior debido al ahorro de energía que no sólo supone un recorte en la factura de la luz, sino que también hace que sea más sostenible y más respetuoso con el medioambiente. Además del ahorro de energía, el LED tiene una mayor producción por lo que se alza como el método con la mejor relación calidad precio. Es decir, menos gasto y mayor rendimiento.

Ilustración 3-7: Iluminación LED para cultivo de interior

Ventajas

Las ventajas son numerosas y variadas. La placa LED puede funcionar durante 11 años en plenas condiciones y sin perder potencia lumínica, mientras que las bombillas de sodio tradicionales comienzan a perder potencia ya a partir del primer cultivo. Otra ventaja principal es el menor consumo.

Además, otra ventaja muy importante es que no desprende calor por lo que se ahorra en sistemas de ventilación. Es decir, con el sodio las temperaturas del interior suben demasiado y es necesario tener un sistema de ventilación en constante funcionamiento. Con el LED la temperatura sube muy poco y no será necesario tener la ventilación activada todo el tiempo. Incluso permite realizar más cultivos en zonas donde el clima es cálido.

Inconvenientes

Aunque no son tan numerosos como las ventajas, se pueden encontrar una serie de inconvenientes a tener en cuenta.

Se ha hablado de la ventaja de que apenas desprendan calor pero en contra partida puede ser una desventaja ya que en zonas frías será necesario emplear calefacción que suba la temperatura de cultivo.

Hay que tener en cuenta también que una temperatura más baja equivale a una mayor humedad por lo que si existe exceso de humedad habrá que instalar un deshumidificador.

Por último, mencionar que algunas placas LED tienen un precio más elevado que el de las placas de iluminación convencionales. Esto hará que se tenga que estudiar si el ahorro en la factura eléctrica compensa la inversión en iluminación LED para según qué cultivos.

3.6. Arduino Nano

La Arduino Nano tiene funcionalidades similares a las de Arduino Duemilanove pero con un paquete diferente. La Nano, basada en el microcontrolador ATmega328 incorpora el modelo ATmega328P, el misma que la tarjeta Arduino Uno. La principal diferencia entre ellos es que el Arduino Uno se presenta en forma de PDIP (*Plastic Dual-In-line Package*) con 30 pines y Arduino Nano está disponible en TQFP (*plastic quad flat pack*) con 32 pines. Los 2 pines adicionales de la placa Arduino Nano sirven para las funcionalidades del ADC, mientras que UNO tiene 6 puertos de tipo ADC, la tarjeta Nano tiene 8 puertos de este.

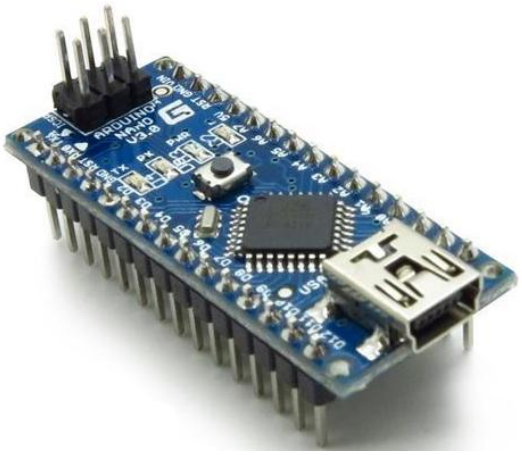

Ilustración 3-8: Placa Arduino Nano

3.6.1. Hardware

Es una placa microcontroladora pequeña, compatible, flexible y fácil de usar desarrollada por Arduino. Esta tarjeta viene con exactamente la misma funcionalidad que Arduino UNO, pero en tamaño más reducido. Además, viene con una tensión de funcionamiento de 5 V, sin embargo, la tensión de entrada puede variar de 7 a 12 V.

La placa Arduino Nano contiene 14 pines digitales y 8 pines analógicos. Cada uno de estos pines digitales y analógicos tiene asignadas múltiples funciones, pero su función principal debe configurarse como entrada o salida. Actúan como pines de entrada cuando están interconectados con los sensores, pero si está conduciendo alguna carga, se recomienda utilizarlos como salida.

En resumen las características técnicas de la tarjeta Arduino Nano son:

- Microcontrolador Arduino ATmega328.
- Arquitectura, AVR.
- Voltaje: 5 V.
- Memoria flash: 32 KB de los cuales 2 KB son utilizados por *bootloader*.
- SRAM: 2 KB.
- Velocidad del reloj: 16 MHz.
- Pines de E/S analógicas: 8.
- EEPROM: 1 KB.
- Corriente continua por pin entrada salida: 40 mA (Pines de E/S).
- Voltaje de entrada, 7-12 V.
- Pines de E/S digitales: 22.
- Salida PWM: 6.

- Consumo de energía: 19 mA.
- Tamaño de la placa de circuito impreso: 18x45 mm.
- Peso: 7g

3.6.2. Software

La placa Arduino Nano está preparada para ser usada con el entorno Arduino. Este Entorno de Desarrollo Integrado funciona tanto offline como online. Como ya se ha comentado y se comentará en otras tarjetas propuestas, poder usar el entorno de Arduino nos hace disponer de una herramienta sencilla y manejable.

Además, no se requieren arreglos previos para hacer funcionar la placa. Todo lo que se necesita es la placa, un cable mini USB y el software Arduino IDE instalado en el ordenador. El cable USB se utiliza para transferir el programa del ordenador a la placa. No se requiere un programador por separado para compilar y grabar el programa, ya que esta placa viene el mismo incorporado.

3.6.3. Arduino Nano Pinout

En la siguiente imagen (*Ilustración 3-11*) se puede observar la distribución de los pines de entrada y salida de la placa Arduino Nano.

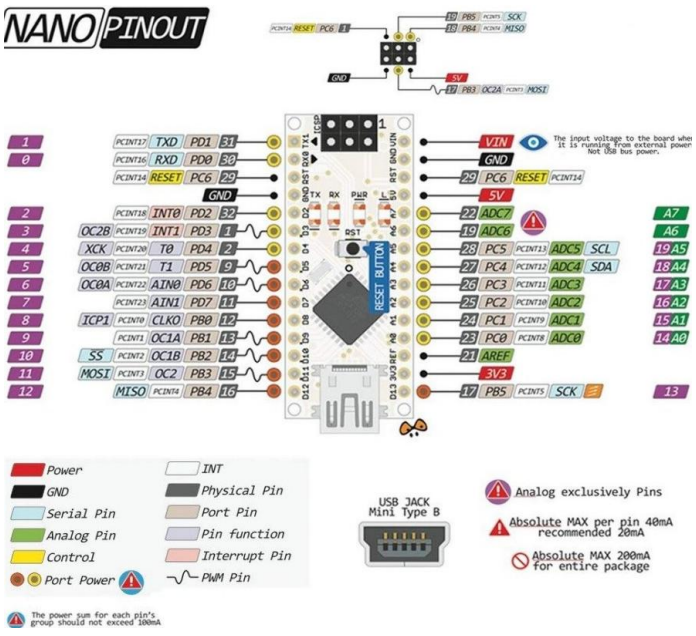

Ilustración 3-9: Diagrama de pines de la tarjeta Arduino Nano

3.7. ESP32

El ESP32 es un SoC (*System on Chip*) diseñado por la compañía china Espressif [Espressif] y fabricado por TSMC. Un microcontrolador de bajo coste y consumo de energía que además integra en un único chip un procesador Tensilica Xtensa de doble núcleo de 32 bits a 160 MHz (con posibilidad de hasta 240 MHz), conectividad WiFi y Bluetooth de modo dual. El ESP32 es el sucesor del SoC ESP8266.

El ESP32 añade muchas funciones y mejoras como mayor potencia, Bluetooth 4.0, encriptación por hardware, sensor de temperatura, sensor hall, sensor táctil, reloj de tiempo real (RTC), más puertos, más buses, etc.

Hay que resaltar el gran potencial para elaborar todo tipo de proyectos, sobre todo por su capacidad de comunicación, ocupando un lugar destacado en aplicaciones de IoT (*Internet of Things*).

Ilustración 3-10: Modulo ESP32

3.7.1. Hardware

El ESP32 tiene ventajas muy claras frente a su predecesor como la inclusión de un segundo procesador (posee 2 núcleos). Se le ha añadido la posibilidad de utilizar *Bluetooth Low Energy* (BLE), una característica muy atractiva para los proyectos del mundo de IoT.

Cuenta con más pines de lecturas analógicas a digitales (ADC) y además, se incluyeron dos pines de salida digital a analógica (DAC) integrados, muy útiles para proyectos que requieran de audio.

Al poseer un segundo núcleo, este trabaja únicamente para manejar los eventos de Wi-Fi aunque se le pueden asignar otras tareas específicas.

Otra característica a destacar es que permite utilizar más sensores de lectura analógica sin necesidad de utilizar multiplexores.

Cabe destacar que tiene un diseño bastante robusto capaz de funcionar en entornos industriales y bajo temperaturas de entre -40 °C y +125 °C.

En resumen, las características técnicas de ESP32 son las expuestas a continuación:

- Procesador Xtensa LX6 de 32 bits de doble núcleo.
- Velocidad de 160 MHz (máximo 240 MHz).
- Co-procesador de ultra baja energía.
- Memoria 520 KiB SRAM.
- Memoria flash externa hasta 16MiB.
- Encriptación de la Flash.
- Arranque seguro.
- Pila de TCP/IP integrada.
- Wi-Fi 802.11 b/g/n 2,4 GHz (soporta WFA/WPA/WPA2/WAPI)
- Bluetooth 4.2 BR/EDR y BLE.
- Criptografía acelerada por hardware.
- 32 pins GPIO.
- Conversor analógico digital (ADC) de 12 bits y 18 canales.
- 2 conversores digital analógico (DAC) de 8 bits.
- 16 salidas PWM (LED PWM).
- 1 salida PWM para motores.
- 11 conversor analógico a digital de 10 pin.
- 10x sensores capacitivos (en GPIO).
- 3* UARTs, 4* SPIs, 2* I^2Ss, 2* I^2Cs, CAN bus 2.0
- Controladora host SD/SDIO/CE-ATA/MMC/eMMC.
- Controladora *slave* SDIO/SPI.
- Sensor de temperatura.
- Sensor de efecto Hall.
- Generador de números aleatorios.
- Reloj tiempo real (RTC).
- Controlador mando a distancia infrarrojos (8 canales).

3.7.2. Software

En cuanto a lenguajes de programación tenemos varias opciones. Es posible emplear el IDE de Arduino, instalar MicroPython, RTOS, Mongoose OS o Espruino.

IDE de Arduino

Arduino es una comunidad tecnológica y compañía dedicada a la producción de hardware libre. Esta empresa se dedica a diseñar y manufacturar diferentes tipos de placas de desarrollo de hardware y software. Éstas se encuentran compuestas por circuitos impresos que son los que integran su microcontrolador. Por lo tanto, a la hora de hablar de estas placas hay que hacer mención del IDE de Arduino.

Arduino IDE es una aplicación multiplataforma (para Windows, macOS, Linux) que está escrita en el lenguaje de programación Java. Se utiliza para escribir y cargar programas en placas compatibles con Arduino, pero también, con la ayuda de núcleos de terceros, se puede usar con placas de desarrollo de otros proveedores. El código fuente para el IDE se publica bajo la Licencia Pública General de GNU, versión 2.

Además, admite los lenguajes C y C ++ utilizando reglas especiales de estructuración de códigos. El IDE de Arduino suministra una biblioteca de software del proyecto Wiring, que proporciona muchos procedimientos comunes de E/S. El código escrito por el usuario solo requiere dos funciones básicas, para iniciar el boceto y el ciclo principal del programa, que se compilan y vinculan con un apéndice de programa *main()* en un ciclo con el GNU *toolchain*, que también se incluye.

Hay que añadir que el IDE de Arduino también emplea el programa *avrdude* para convertir el código ejecutable en un archivo de texto en codificación hexadecimal que se carga en la placa Arduino mediante un programa de carga en el firmware de la placa.

MicroPython

Es una implementación software del lenguaje de programación Python 3, escrita en C, y que está optimizada para poder ejecutarse en un microcontrolador.

MicroPython es un compilador completo del lenguaje Python y un motor e intérprete en tiempo de ejecución, que funciona en el hardware del microcontrolador. Al usuario se le presenta una línea de órdenes interactiva (el REPL) que soporta la ejecución inmediata de órdenes. Se incluye una selección de bibliotecas fundamentales de Python: MicroPython incluye módulos que permiten al programador el acceso al hardware en bajo nivel.

Fue creado originalmente por el programador y físico australiano Damien George, después de una exitosa campaña de Kickstarter que apoyó el proyecto en 2013. Aunque durante la campaña original de Kickstarter se lanzó MicroPython en conjunción con la placa de microcontrolador PyBoard, en la actualidad MicroPython

soporta un amplio número de arquitecturas basadas en ARM. Desde entonces MicroPython se ha conseguido ejecutar en plataformas basadas en Arduino, ESP8266, ESP32, e Internet de las cosas.

RTOS

Un sistema operativo de tiempo real es un sistema operativo que ha sido desarrollado para aplicaciones de tiempo real en sistemas embebidos. Antes es necesario aclarar su definición:

- **Sistema operativo:** es un programa que gestiona el hardware de un dispositivo y permite que subprogramas se ejecuten mediante los servicios que proporciona. Simplifica el desarrollo de aplicaciones específicas, ya que gestiona toda la parte de control del hardware y no hay necesidad de programar a bajo nivel.
- **Tiempo real:** los sistemas en tiempo real son capaces de gobernar varios periféricos en un determinado tiempo.
- **Sistemas embebidos:** es un anglicismo de *embedded systems,* también se llaman sistemas empotrados. Son sistemas diseñados para realizar unas tareas concretas.

Por lo tanto a este tipo de sistemas se le exige corrección en sus respuestas bajo ciertas restricciones de tiempo. Si no las respeta, se dirá que el sistema ha fallado. Para garantizar el comportamiento correcto en el tiempo requerido se necesita que el sistema sea predecible.

La ventaja de usar RTOS frente a los métodos de programación tradicionales, reside en la simplificación de la gestión de las tareas.

Mongoose

Mongoose es una biblioteca de JavaScript que le permite definir esquemas con datos fuertemente tipados. Una vez que se define un esquema, Mongoose le permite crear un modelo basado en un esquema específico. Un modelo de mangosta se asigna a un documento MongoDB a través de la definición del esquema del modelo.

Por lo tanto, Mongoose OS es un framework de desarrollo de firmware para el Internet de las cosas (*IoT*) disponible bajo la licencia Apache 2.0. Es compatible con microcontroladores conectados de bajo consumo como son: ESP32, ESP8266, TI CC3200, STM32. Su propósito es ser un entorno completo para hacer prototipos, desarrollar y administrar dispositivos conectados. Se pueden crear proyectos en

lenguaje mJS "*Javascript*" o C/C++,además de que cuenta con una interface web como IDE o desde terminal de comandos.

Espruino

Es un intérprete de JavaScript de código abierto que soporta diferentes microcontroladores. Está diseñado para dispositivos con pequeñas cantidades de memoria

Es compatible con placas de funcionalidad especial Arduino y, además de las placas oficiales, Espruino se ejecuta en aproximadamente en 40 otros tipos de placas de desarrollo incluyendo el ESP8266 y ESP32.

3.7.3. Protocolos ESP32

El ESP32 puede comunicarse mediante los protocoles SPI, I^2C, UART o Host SD.

Protocolo SPI

SPI es un acrónimo para referirse al protocolo de comunicación serial *Serial Peripherical Interface*. El es un protocolo síncrono que trabaja en modo full dúplex para recibir y transmitir información, permitiendo que dos dispositivos pueden comunicarse entre sí al mismo tiempo utilizando canales diferentes o líneas diferentes en el mismo cable. Al ser un protocolo síncrono el sistema cuenta con una línea adicional a la de datos encarga de llevar el proceso de sincronismo.

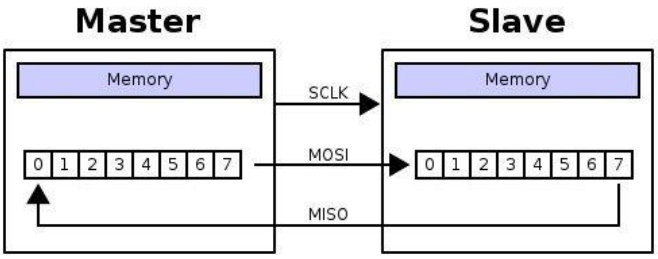

Ilustración 3-11: Estructura general del protocolo SPI

Dentro de este protocolo se define un maestro que será aquel dispositivo encargado de transmitir información a sus esclavos.

Los esclavos serán aquellos dispositivos que se encarguen de recibir y enviar información al maestro. El maestro también puede recibir información de sus dispositivos tipo esclavo.

Para que este proceso se haga realidad es necesario la existencia de dos registros de desplazamiento, uno para el maestro y uno para el esclavo respectivamente. Los registros de desplazamiento se encargan de almacenar los bits de manera paralela para realizar una conversión paralela a serial para la transmisión de información.

Como ya se ha comentado, el protocolo SPI es un protocolo síncrono por lo que la sincronización y transmisión de datos se realiza mediante cuatro señales:

- **SCLK** *(Clock)*: Es el pulso que marca la sincronización. Con cada pulso de este reloj, se lee o se envía un bit.
- **MOSI** *(Master Output Slave Input)*: Salida de datos del maestro y entrada de datos al esclavo. También llamada SIMO.
- **MISO** *(Master Input Slave Output)*: Salida de datos del esclavo y entrada al maestro. También conocida por SOMI.
- **SS/Select**: Para seleccionar un esclavo, o para que el maestro le diga al esclavo que se active. También llamada SSTE.

En resumen, es un protocolo que ha ganado importancia en el entorno de la industria como sistema de comunicación de corta distancia. Permite alcanzar velocidades altas de transmisión y que se diseñó pensando en comunicar distintos periféricos con un único microcontrolador.

Protocolo I^2C

I^2C significa Circuito Interintegrado (*Inter-Integrated Circuit*). El protocolo I^2C toma e integra lo mejor de los protocolos SPI y UART. Con el protocolo I^2C podemos tener a varios maestros controlando uno o múltiples esclavos pero sólo uno puede ser el maestro cada vez. El cambio de maestro supone una alta complejidad, por lo que no es algo frecuente. Aun así, puede ser de gran ayuda cuando se van a utilizar varios microcontroladores para almacenar un registro de datos hacia una sola memoria o cuando se va a mostrar información en una sola pantalla.

Fue desarrollado en 1982 por Philips Semiconductors (hoy NXP Semiconductors, parte de Qualcomm). Se utiliza internamente para la comunicación entre diferentes partes de un circuito, por ejemplo, entre un controlador y circuitos periféricos integrados.

El sistema original fue desarrollado por Philips a principios de 1980 con el fin de controlar varios chips en televisores de manera sencilla. Desde mediados de 1990 el I^2C también es utilizado por algunos competidores para designar los sistemas compatibles I^2C Philips, incluyendo Siemens AG (posteriormente Infineon Technologies AG), NEC, STMicroelectronics, Motorola (Freescale más adelante), Intersil, etc. Hay un total de mil circuitos integrados diferentes de más de 50 fabricantes (según datos de 2014).

El protocolo I^2C utiliza sólo dos vías o cables de comunicación, así como también lo hace el protocolo UART.

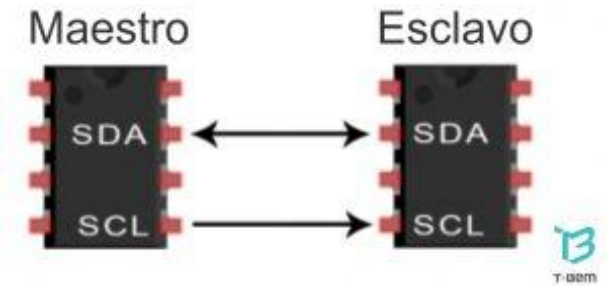

Ilustración 3-12: Estructura general del I^2C

El bus I^2C es síncrono. El maestro proporciona una señal de reloj, que mantiene sincronizados a todos los dispositivos del bus.

De esta forma, se elimina la necesidad de que cada dispositivo tenga su propio reloj, de tener que acordar una velocidad de transmisión y mecanismos para mantener la transmisión sincronizada (como usando UART).

Además, I^2C es un protocolo de comunicación serial. El protocolo I^2C envía información a través de una sola vía de comunicación y dicha información es enviada bit por bit de forma coordinada.

Protocolo UART

UART, son las siglas en inglés de *Universal Asynchronous Receiver-Transmitter*, en español: Transmisor-Receptor Asíncrono Universal, es el dispositivo que controla los puertos y dispositivos serie. Se encuentra integrado en la placa base o en la tarjeta adaptadora del dispositivo.

Un UART dual, o DUART, combina dos UART en un solo chip. Existe un dispositivo electrónico encargado de generar la UART en cada puerto serie. La mayoría de las computadoras modernas utilizan el chip UART 16550, que soporta velocidades de transmisión de hasta 921,6 Kbps. Las funciones principales de chip UART son: manejar las interrupciones de los dispositivos conectados al puerto serie y convertir los datos en formato paralelo, transmitidos al bus de sistema, a datos en formato serie, para que puedan ser transmitidos a través de los puertos y viceversa.

Por lo tanto, se encarga de leer datos cuando llegan, generar y gestionar interrupciones, enviar datos y gestionar los tiempos de bit.

El UART normalmente no genera directamente o recibe las señales externas entre los diferentes módulos del equipo. Usualmente se usan dispositivos de interfaz separados para convertir las señales de nivel lógico del UART hacia y desde los niveles de señalización externos.

Protocolo Host SD

El protocolo de configuración dinámica de host es un protocolo de red de tipo cliente/servidor mediante el cual un servidor DHCP asigna dinámicamente una dirección IP y otros parámetros de configuración de red a cada dispositivo en una red para que puedan comunicarse con otras redes IP. Este servidor posee una lista de direcciones IP dinámicas y las va asignando a los clientes conforme estas van quedando libres, sabiendo en todo momento quién ha estado en posesión de esa IP, cuánto tiempo la ha tenido y a quién se la ha asignado después.

Puede configurar los dispositivos SD-WAN como servidores DHCP o agente de retransmisión DHCP. La función del servidor DHCP permite que los dispositivos de la misma red que la interfaz LAN/WAN del dispositivo SD-WAN obtengan su configuración IP desde el dispositivo SD-WAN. La función de retransmisión DHCP permite que los dispositivos SD-WAN reenvíen paquetes DHCP entre el cliente DHCP y el servidor.

3.7.4. ESP32 Pinout

En la *Ilustración 3-15* podemos observar la distribución de los pines I/O de la placa de desarrollo, fundamental y necesario a la hora de trabajar con esta placa.

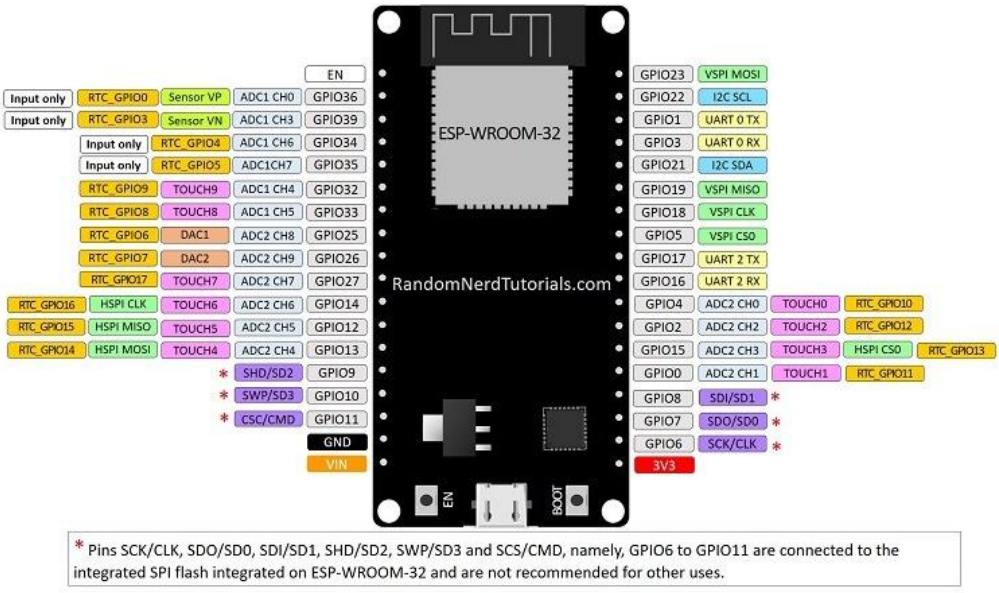

Ilustración 3-13: Diagrama de pines ESP32

3.8. Aplicación Android

Android es un sistema operativo desarrollado por Google basado en Kernel de Linux y otros software de código abierto, y en una primera instancia pensado para teléfonos móviles, al igual que iOS, Symbian y Blackberry OS. Como está basado en Linux, lo hace un núcleo de sistema operativo libre, gratuito y multiplataforma.

Inicialmente fue desarrollado por Android Inc., empresa que Google respaldó económicamente y que adquirió en 2005. Fue presentado en 2007 junto con la fundación del Open Handset Alliance (un consorcio de compañías de hardware, software y telecomunicaciones) para avanzar en los estándares abiertos de los dispositivos móviles. El código fuente principal de Android se conoce como Android Open Source Project (AOSP), que se licencia principalmente bajo la Licencia Apache.

Este sistema permite programar aplicaciones en una variación de Java llamada Dalvik. Hasta la versión 4.4.3, Android utiliza Dalvik como máquina virtual con la compilación "justo a tiempo" (*Just In Time*) para ejecutar Dalvik *dex-code* (Dalvik ejecutable), que es una traducción de Java *bytecode*. Siguiendo el principio JIT, además de la interpretación de la mayoría del código de la aplicación, Dalvik realiza la compilación y ejecución nativa de segmentos de código seleccionados que se ejecutan con frecuencia cada vez que se inicia una aplicación. Android 4.4 introdujo el ART (*Android Runtime*) como un nuevo entorno de ejecución, que compila el Java *bytecode* durante la instalación de una aplicación. Este se convirtió en la única opción en tiempo de ejecución en la versión 5.0.

Es capaz de proporcionar todas las interfaces necesarias para desarrollar aplicaciones que accedan a las funciones del teléfono (como el GPS, las llamadas, etc.) de una forma muy sencilla en un lenguaje de programación muy conocido como es Java.

Su sencillez y la cantidad de herramientas de programación existentes gratuitas hacen que una de sus principales características sea la cantidad de aplicaciones disponibles.

Como es completamente libre, esto supone una de las mejores características ya que ni para programar en este sistema ni para incluirlo en un teléfono hay que pagar nada. Y esto lo hace muy popular entre fabricantes y desarrolladores, ya que los costes para lanzar un teléfono o una aplicación son muy bajos.

Aplicación móvil Android

Una aplicación móvil o *app* (acortamiento del inglés *application*), es una aplicación informática diseñada para ser ejecutada en teléfonos inteligentes, tabletas y otros dispositivos móviles. Este tipo de aplicaciones permiten al usuario realizar un variado conjunto de tareas (profesionales, de ocio, educativas, de acceso a servicios, etc.), facilitando las gestiones o actividades a desarrollar.

Por lo general, se encuentran disponibles a través de ciertas plataformas de distribución, o por intermedio de las compañías propietarias de los sistemas operativos móviles tales como Android, iOS, BlackBerry OS, Windows Phone, entre otros. Existen aplicaciones móviles gratuitas y otras de pago.

Estas aplicaciones están escritas en un lenguaje de programación compilado, y su funcionamiento y recursos se encaminan a aportar una serie de ventajas:

- Un acceso más rápido y sencillo a la información necesaria sin necesidad de los datos de autenticación en cada acceso.
- Un almacenamiento de datos personales que, a prioridad, es de una manera segura.
- Una gran versatilidad en cuanto a su utilización o aplicación práctica.
- La atribución de funcionalidades específicas.
- Mejorar la capacidad de conectividad y disponibilidad de servicios y productos.

Aspectos principales

Para programar aplicaciones Android es posible usar los lenguajes Kotlin, Java y C++. Las herramientas de Android SDK compilan tu código, junto con los archivos de recursos y datos, en un APK: un *paquete de Android*, que es un archivo de almacenamiento con el sufijo *.apk*. Un archivo APK incluye todos los contenidos de una aplicación de Android y es el archivo que usan los dispositivos con tecnología Android para instalar la aplicación.

Además, suelen estar protegidas mediante una serie de características de seguridad tales como:

- El sistema operativo Android es un sistema Linux multiusuario en el que cada aplicación es un usuario diferente.
- De forma predeterminada, el sistema le asigna a cada aplicación un ID de usuario de Linux único (sólo el sistema utiliza el ID y la aplicación lo desconoce). El sistema establece permisos para todos los archivos en una aplicación de modo que solo el ID de usuario asignado a esa aplicación pueda acceder a ellos.

- Cada proceso tiene su propia máquina virtual (VM), por lo que el código de una aplicación se ejecuta de forma independiente de otras aplicaciones.
- De forma predeterminada, cada aplicación ejecuta su propio proceso de Linux. El sistema Android inicia el proceso cuando se requiere la ejecución de alguno de los componentes de la aplicación y, luego, lo cierra cuando el proceso ya no es necesario o cuando el sistema debe recuperar memoria para otras aplicaciones.

De esta manera, el sistema Android implementa el *principio de mínimo privilegio*. Es decir, de forma predeterminada, cada aplicación tiene acceso solo a los componentes que necesita para llevar a cabo su trabajo y nada más. Esto crea un entorno muy seguro, en el que una aplicación no puede acceder a partes del sistema para las que no tiene permiso.

Componentes de la aplicación

Son bloques de creación de una aplicación para Android. Cada uno de ellos es un punto de entrada por el que el sistema o el usuario entra a la aplicación.

Hay cuatro tipo y cada tipo tiene un fin específico y un ciclo de vida característico que define cómo se crea y se destruye el componente.

- **Activities**: punto de entrada de interacción con el usuario. Representa una pantalla individual con una interfaz de usuario.
- **Services**: punto de entrada general que permite mantener la ejecución de una aplicación en segundo plano por diversos motivos. Es un componente que se ejecuta en segundo plano para realizar operaciones de ejecución prolongada o para realizar tareas de procesos remotos. Un servicio no proporciona una interfaz de usuario.
- **Broadcast Receivers**: componente que posibilita que el sistema entregue eventos a la aplicación fuera de un flujo de usuarios habitual, lo que permite que la aplicación responda a los anuncios de emisión de todo el sistema. Dado que los receptores de emisión son otro punto bien definido de entrada a la aplicación, el sistema puede entregar emisiones incluso a las aplicaciones que no estén en ejecución.
- **Content Provider**: administra un conjunto compartido de datos de la aplicación que puedes almacenar en el sistema de archivos, en una base de datos SQLite, en la Web o en cualquier otra ubicación de almacenamiento persistente a la que tenga acceso tu aplicación. A través del proveedor de contenido, otras aplicaciones pueden consultar o modificar los datos si el proveedor de contenido lo permite.

3.9. App Inventor

Entorno de desarrollo de software creado por Google Labs para la elaboración de aplicaciones destinadas al sistema operativo Android. El usuario puede, de forma visual y a partir de un conjunto de herramientas básicas, ir enlazando una serie de bloques para crear la aplicación. El sistema es gratuito y se puede descargar fácilmente de la web. Las aplicaciones creadas con App Inventor están limitadas por su simplicidad, aunque permiten cubrir un gran número de necesidades básicas en un dispositivo móvil.

La plataforma se puso a disposición del público el 25 de diciembre de 2008 y está dirigida a personas que no están familiarizadas con la programación, contando con la ayuda que nos brinda la informática. En la creación de App Inventor, Google se basó en investigaciones previas significativas en informática educativa y sirve para crear páginas.

Por lo tanto, para desarrollar aplicaciones con App Inventor sólo necesitas un navegador web y un teléfono o tablet Android.

Proceso de creación de una aplicación

El proceso de creación de una app con MIT App Inventor consta de 3 fases:

1. Diseñador de pantallas**:** se crean las distintas ventanas o pantallas que contendrá la aplicación. En ellas sitúan sus componentes: imágenes, botones, textos, etc. y se configuran sus propiedades.
2. Editor de bloques: permite programar de forma visual e intuitiva el flujo de funcionamiento del programa utilizando bloques. Cada objeto dispone de unos métodos específicos que es posible invocar personalizando sus parámetros de llamada.
3. Generador de app: al finalizar las fases de diseño y programación, se genera el instalador APK de la aplicación. Se puede obtener un código QR para su descarga temporal desde el móvil o bien el propio archivo APK para descargar, publicar en la nube y/o enviar a otros usuarios/as.

Capítulo 4

HARDWARE

4.1. Introducción

En el presente capítulo se va a proceder a la descripción de los diversos dispositivos que van a estar integrados en este proyecto, así como cada uno de los componentes que se encargarán del funcionamiento de la cámara.

Esta composición de dispositivos y componentes nos permitirá diseñar un sistema electrónico y automático que cumplirán con las especificaciones que requiere dicha cámara, tanto de diseño como de funcionamiento.

Por lo tanto, se expondrán y se detallarán los dispositivos seleccionados así como la lógica de su funcionamiento.

4.2. Componentes del sistema

A continuación se van a citar los principales componentes seleccionados que van a componer físicamente el sistema.

4.2.1. ESP32

El dispositivo principal del sistema va a ser el módulo ESP32 de Espressif. Este módulo es una placa de desarrollo donde ya viene incorporado el chip.

Ilustración 4-1: Módulo ESP32

Este módulo hará de placa principal de control entre la interfaz con el usuario y la tarjeta que llevará a cabo las funciones programadas. Es decir, el ESP32 realizará las tareas principales de gestión de datos y de gestión de recursos.

4.2.2. Arduino Nano

Dentro de las posibles opciones comentadas en el capítulo anterior se ha decidido utilizar el microcontrolador Arduino Nano en el sistema. Esta tarjeta permitirá la lectura y adquisición de datos solicitada por la placa de control ESP32, dejando a esta última con las únicas tareas de gestión de datos y recursos, y comunicación con la interfaz del usuario.

Ilustración 4-2: Arduino Nano ATmega328

4.2.3. Sensor de temperatura y humedad

Como se ha desarrollado en capítulos anteriores, es necesario implementar sensores capaces de medir y registrar la temperatura y la humedad para cumplir los objetivos que deberá cumplir la cámara. Anteriormente se hizo un estudio de las opciones disponibles para ambos parámetros y comparando dichas opciones encontramos la existencia de módulos capaces de medir tanto temperatura como humedad.

Por lo tanto seleccionamos esta opción porque permitirá cumplir los dos requisitos con un único elemento.

4.2.3.1. Opciones disponibles

DHTXX es una familia de sensores que permite realizar la medición simultánea de temperatura y humedad. Estos sensores disponen de un sensor capacitor para medir la humedad, un termistor y un procesador interno que realiza el proceso de medición, proporcionando dicha medición mediante una señal digital. Debido a esto, resulta muy sencillo obtener la medición desde un microprocesador como sería, en este caso, Arduino.

Dentro de esta familia existen dos modelos: el DHT11 y el DHT22 (o AM2302).

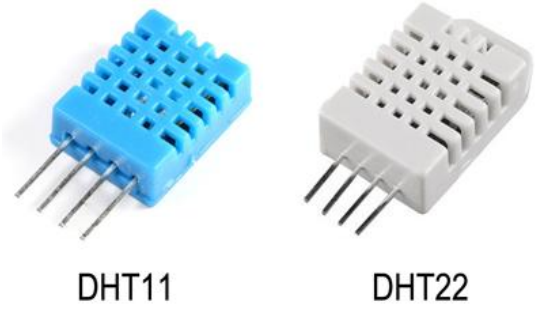

Ilustración 4-3: sensor DHT11 y sensor DHT22

Comparativa DHT11 y DHT22.

Ambos sensores presentan un encapsulado de plástico similar. Se pueden distinguir ambos modelos por el color de los mismos. El DHT11 presenta una carcasa azul, mientras que la carcasa del DHT22 es blanca. Además, en el ámbito del encapsulado, existen tres variantes en el mercado para ambos sensores:

- El sensor suelto, con la carcasa azul o blanca según el modelo y cuatro pines disponibles para conectar. Este formato hace necesario la utilización de resistencias de tipo pull-up.

- El sensor con una placa PCB soldada, con tres pines disponibles para conectar y una resistencia *Pull-Up* (generalmente de 4,7-10 kΩ).
- El mismo formato que el anterior, pero con un condensador de filtrado (generalmente de 100 nF).

En lo que a características técnicas se refiere, el DHT11 es el hermano pequeño de la familia, y cuenta con características técnicas inferiores a las del DHT22. Por lo tanto, el DHT22 es el modelo superior de esta familia de sensores y se verá reflejado en el precio.

En la *tabla 4-1* que se presenta una comparativa de las características técnicas del módulo DHT11 frente al módulo DHT22.

Parámetros	DHT11	DHT22
Alimentación	3 V ≤ Vcc ≤ 5 V	3,3 V ≤ Vcc ≤ 6 V
Señal de salida	Digital	Digital
Rango de Tª	De 0 a 50 °C	De -40 °C a 80 °C
Precisión de temperatura	±2 °C	< ±0,5 °C
Resolución de temperatura	0,1 °C	0,1 °C
Precisión de humedad	De 20% a 90% RH	De 0 a 100% RH
Resolución de humedad	1% RH	0,1% RH
Tiempo de respuesta	1s	2s

Tabla 4-1: Comparativa DHT11 - DHT22

Por lo tanto, se puede llegar a la conclusión de que el DHT11 es un sensor limitado que podemos usar con fines de formación, pruebas, o en proyectos que realmente no requieran una medición precisa. En cambio, el DHT22 tiene unas características más aceptables y por lo que se puede utilizar en proyectos reales de monitorización o registro, que requieran una precisión media.

4.2.3.2. Sensor DHT22

Dado lo expuesto en la sección anterior, se ha seleccionado el módulo DHT22 para medir temperatura y humedad. Su relativo bajo precio (varía entre 2,50 € - 7 €), su precisión es alta y su rango de operación coincidente con las condiciones ambientales en las que va a trabajar la cámara, lo hacen el sensor más adecuado para este proyecto.

El DHT22 sigue siendo un sensor digital de temperatura y humedad relativa de buen rendimiento. Integra un sensor capacitivo de humedad y un termistor para medir el aire circundante como todos los de la familia DHTXX, y muestra los datos mediante una señal digital en el pin de datos (no posee salida analógica). Este tipo de sensor es frecuentemente utilizado en aplicaciones de control automático de temperatura, aire acondicionado, monitoreo ambiental en agricultura, etc.

Su uso en las plataformas Arduino/Raspberry Pi/Nodemcu es muy sencillo tanto a nivel de software como hardware, por lo que es muy conveniente para este proyecto ya que se trabajará con Arduino. A nivel de software se dispone de librerías para Arduino con soporte para el protocolo "*Single bus*". En cuanto al hardware, sólo es necesario conectar el pin VCC de alimentación a 3,3-5 V, el pin GND a tierra y el pin de datos a un pin digital en el Arduino. Si se deseara conectar varios sensores DHT22 a un mismo Arduino, cada sensor debe tener su propio pin de datos.

En la *ilustración 4-4*, podemos observar las dimensiones del sensor de DHT22 del fabricante Aosong [Aosong].

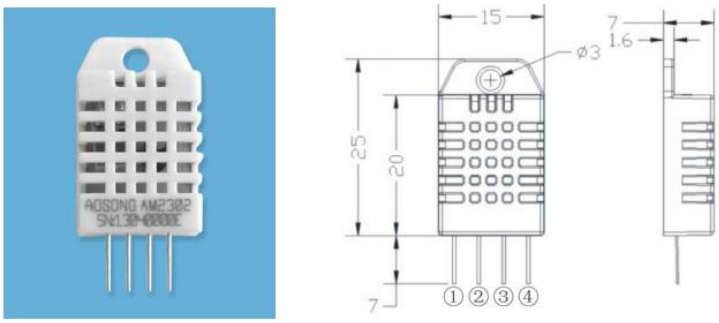

Ilustración 4-4: Dimensiones del sensor DHT22

En concreto, se ha seleccionado el módulo DHT22 integrado en una placa PCB para mayor facilidad en la conexión y el ahorro de soldar resistencias de tipo *Pull-Up*. En la *ilustración 4-5* se observa el sensor DHT22 integrado en una placa PCB y su *pinout*. En la *tabla 4-2* se tiene la asignación de estos pines.

Ilustración 4-5: sensor DHT22 integrado en una PCB y pinout

Pin	Nombre	Descripción
1	GND	Tierra (0 V)
2	VCC	Potencia (3,3 V-6 V)
3	DAT	Canal datos del sensor

Tabla 4-2: Asignación de pines

A continuación se exponen las características eléctricas. Tabla obtenida del modelo Aosong [Aosong].

Parameter	Condition	min	typ	max	Unit
Voltage		3.3	5	5.5	V
Power consumption [4]	Dormancy	10	15		μA
	Measuring		500		μA
	Average		300		μA
Low level output voltage	I_{OL} [5]	0		300	mV
High output voltage	Rp<25 kΩ	90%		100%	VDD
Low input voltage	Decline	0		30%	VDD
Input High Voltage	Rise	70%		100%	VDD
Rpu [6]	VDD = 5V VIN = VSS	30	45	60	kΩ
Output current	turn on		8		mA
	turn off	10	20		μA
Sampling period		2			S

Ilustración 4-6: características eléctricas del sensor DHT22

4.2.4. Sistema de refrigeración

Para el sistema de refrigeración de la cámara se ha decidido utilizar un kit de refrigeración Peltier basado en efecto termoeléctrico.

Efecto y célula Peltier

El componente principal de este kit es la célula Peltier. Este dispositivo semiconductor está basado en efectos termoeléctricos y permite refrigerar de forma muy rápida. Son bastante usadas en diferentes sectores de la industria para refrigerar y esto es debido a que tiene muchas ventajas frente a otros sistemas de enfriamiento tradicionales. Por ejemplo, en el caso de utilizar un dispensador de agua para refrigerar, este se dedica a enfriar el depósito de agua para que se mantenga fresca. Esto conlleva el uso de un deshumidificador que enfría el aire entrante, se condensará la humedad y goteará en el depósito de condensación. En este ejemplo, utilizando la célula Peltier, evitaríamos goteos por condensación.

Como se explicó en capítulos anteriores, los efectos termoeléctricos son aquellos que convierten una diferencia de temperatura en voltaje eléctrico o viceversa. Esto se consigue usando termopares o unos tipos de materiales muy concretos, normalmente semiconductores. En este proyecto se va a utilizar el efecto Peltier donde uno de los lados de la célula se va a calentar y el otro se enfriará.

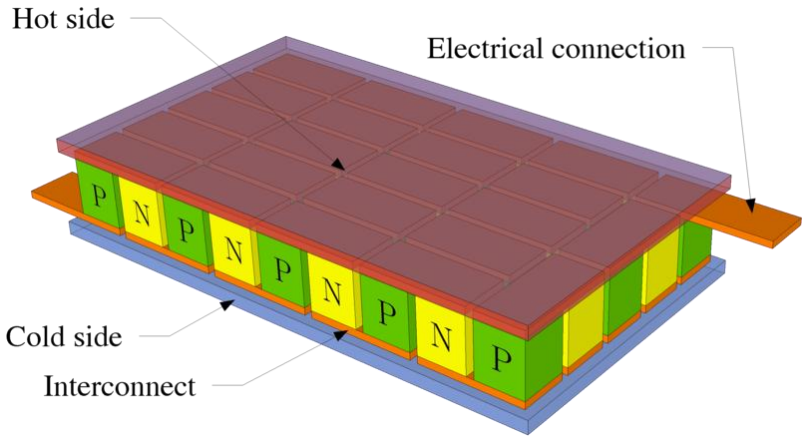

Ilustración 4-7: funcionamiento del efecto Peltier

Una célula Peltier de una sola etapa puede generar una diferencia de temperatura entre sus partes de hasta 70 °C. Por lo tanto, si se mantiene la parte caliente refrigerada, más capacidad de enfriamiento tendrá y ese calor absorbido será proporcional a la corriente proporcionada y al tiempo.

Ventajas y desventajas de la célula Peltier.

En la tabla 4-3 se comparan las ventajas y desventajas de usar la célula Peltier.

Ventajas	Desventajas
No tiene partes móviles, por lo que no requiere mantenimiento y es más fiable.	Solo se puede disipar una cantidad limitada de flujo de calor.
No usa compresores ni gas CFC (clorofluorocarbonos) contaminantes.	No es eficiente energéticamente hablando en comparación con los sistemas de compresión de gas. No obstante, los nuevos avances hacen que sea cada vez más eficiente.
Se puede controlar fácilmente la temperatura y con mucha precisión, hasta fracciones de grado variando la corriente aplicada.	
Tamaño pequeño, aunque se pueden fabricar con diferentes tamaños.	
Tiene una larga vida de hasta 100.000 horas, frente a lo que aportan algunos refrigeradores mecánicos.	

Tabla 4-3: ventajas y desventajas de la célula Peltier

Propiedades.

Una placa Peltier puede tener un precio bastante bajo (1 € - 2 €), así que es un componente muy asequible. Esta placa tiene unas dimensiones de 40x40x3 mm y contiene 127 pares semiconductores en su interior. La potencia eléctrica es de 60 W y su tensión nominal de alimentación de 12 V y corriente nominal de 5 A.

Con ella se puede generar diferencias máximas de temperatura entre sus caras de entre 65 °C y 70 °C. Puede funcionar entre los -55 °C y los 83 °C sin dañarse. Si se mantienen los valores recomendados, puede durar incluso 200.000 horas de trabajo.

La eficiencia de este modelo es de unos 12 - 15 W de calor extraído, eso es una eficiencia en torno al 20 o 25% teniendo en cuenta que consume unos 60 W. Sin embargo hay que tener en cuenta que el valor también se verá muy influenciado por la temperatura ambiente.

Esta célula normalmente se usa tal cual, es decir, forma parte de un kit para refrigeración que ayuda y mejora las prestaciones de la célula y aumenta la refrigeración.

Kit de refrigeración Peltier.

El kit está compuesto, además de por la célula Peltier, por disipadores y ventiladores que ayudan a expulsar el calor absorbido.

En la *Ilustración 4-7* se puede ver un kit de refrigeración Peltier simple aunque se pueden encontrar compuestos de varios juntos como el que se va a ilustrar a continuación.

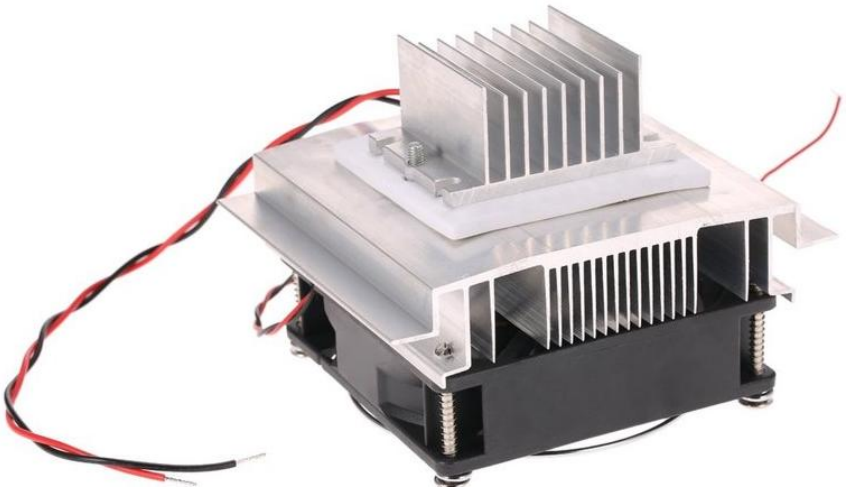

Ilustración 4-8: kit de refrigeración Peltier

Teniendo en cuenta las características de la célula Peltier y ya que cumple las especificaciones de temperatura que se describieron para la cámara, estos kits de refrigeración son los más apropiados para este proyecto.

4.2.5. Sistema de calefacción

Debido a los rangos de temperatura, el tamaño del habitáculo y al clima interno al que estará sometido la cámara, se ha seleccionado como sistema de calefacción el uso del equipo Peltier.

Como ya se ha comentado en secciones anteriores, Peltier es capaz de calentar por un lado y enfriar por el otro cuando se le aplica una diferencia de potencial. Para este caso, haremos uso del lado que genera calor para calentar la cámara cuando el habítaculo así lo requiera.

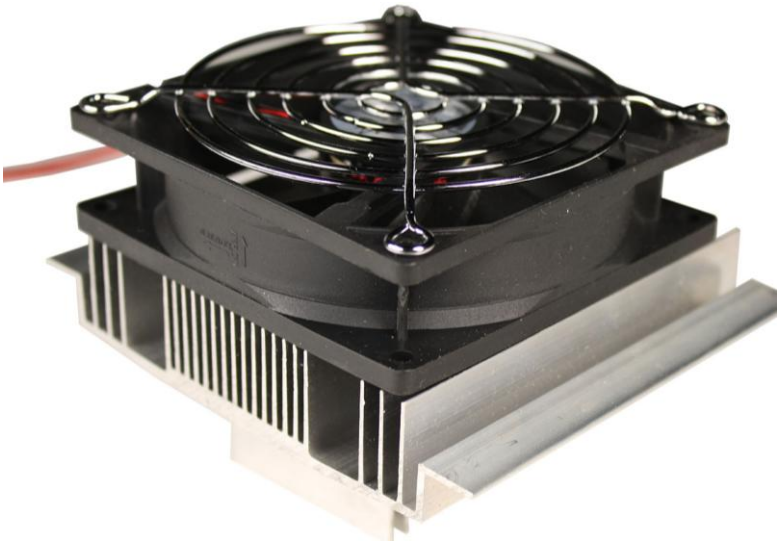

Ilustración 4-9: sección para calefacción Peltier

4.2.6. Sistema de deshumidificación

Atendiendo a las características climáticas requeridas dentro del habitáculo y los rangos habituales de humedad necesarios, se ha decidido deshumidificar la cámara extrayendo la humedad mediante flujos de aire.

Este sistema de deshumidificación estará compuesto por un equipo de ventiladores que, atendiendo a la humedad requerida en el interior de la cámara, se accionarán cuando la situación lo requiera extrayendo mediante los mencionados flujos de aire la humedad en exceso del interior.

Ilustración 4-10: ventilador para extracción de humedad

4.2.7. Iluminación LED

Para recrear el espectro de luz que reciben las plantas del Sol, necesario para su crecimiento, se ha seleccionado el uso de iluminación LED.

El cultivo interior LED da la posibilidad de manipular el espectro de luz que reciben las plantas que están creciendo bajo focos LED y es capaz de aumentar la producción de los cultivos sin afectar la calidad respecto a otras formas de iluminación tradicional. Con este avance tecnológico resulta aconsejable sustituir los sistemas de iluminación fotosintética de lámparas incandescentes, por LEDs que emiten espectros de manera separada. Los dispositivos LED consumen hasta 5 veces menos energía, poseen una vida útil mucho mayor y no generan exceso de calor dañino a las plantas. Además, en el caso de los LEDs rojos, hay una ventaja extra y es que repelen los insectos disminuyendo el uso de tóxicos en los cultivos.

Para el proceso de fotosíntesis, todas las plantas utilizan rangos de longitud de onda de luz de 400 nm hasta los 700nm. El espectro de la radiación recibida puede afectar tanto el crecimiento de la planta, como su floración; y en el caso de las plantas con aplicaciones medicinales, puede afectar el sabor, olor, etc. Esta irradiación fotosintética responsable de la excitación de la clorofila, es mayor en la franja roja del espectro que en la azul, de modo que los vegetales emplean de forma más eficiente la radiación de la región del rojo. La mayor parte de la luz solar que captan las plantas es convertida en calor y solamente la luz roja y azul es esencial para su crecimiento. El cultivo interior LED permite eliminar aquellas longitudes de onda que son inactivas para la fotosíntesis.

Básicamente los LEDs que necesitaría una planta para vivir y desarrollarse con plenitud serian los siguientes:

- **LEDs blancos**: Simulan la luminosidad.
- **LEDs rojos**: Encargados de aportar rayos infrarrojos necesarios para el florecimiento.
- **LEDs azules:** Aportan rayos UV necesarios para el crecimiento.

Los LED han alcanzado una gran potencia y capacidad. Gracias a su optimización han resultado una fuente de irradiación realmente económica para el crecimiento de las plantas, perfectos para usarse en invernaderos, cámaras de crecimiento controlado (cultivos hidropónicos o aeropónicos) o en camas de cultivo en suelo o sustrato.

La instalación de LEDs de colores tiene otro beneficio: las luces se pueden combinar dependiendo de la especificidad del cultivo, o incluso modificar el espectro de luz acompañando el proceso de crecimiento y floración.

Ilustración 4-11: iluminación LED en cultivo de interior

En este proyecto se ha decidido usar tiras LED programables que, además de incorporar protección y resistencia al agua, nos permiten mediante su controlador programar vía Arduino la secuencia de LEDs y el brillo necesario. También destacar que permiten cortar dicha tira hasta la longitud que necesitemos por lo que es adaptable a las medidas de la cámara.

En concreto se ha elegido el modelo de tira LEDs WS2812B de la marca NooElec. WS2812B es el driver o controlador que permitirá el control y la programación de sus LEDs. Dispone de 60 LEDs, clase de eficiencia energética A y un rango de protección

IP68, el cual indica que esta tira LED aguanta hasta un nivel 6 de polvo (significa que no entra el polvo bajo ninguna circunstancia) y hasta un nivel 8 de agua (el equipo aguanta ante una inmersión completa y continua en agua, es decir, no debe entrar agua).

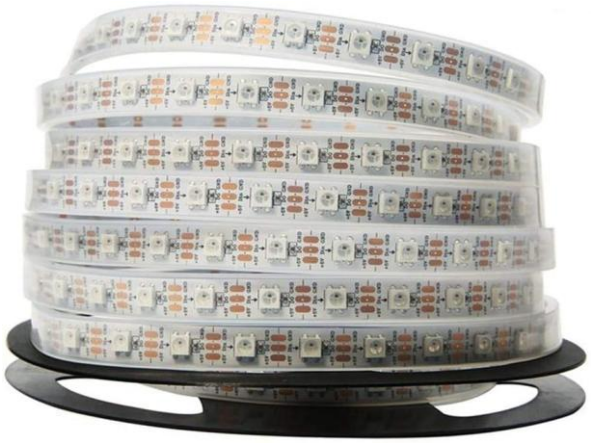

Ilustración 4-12: tira LED para cultivo de interior

4.2.8. Alimentación

La alimentación se realizará a partir de dos fuentes de alimentación de CC (corriente continua).

Una fuente de alimentación será un convertidor AC - DC de 5 V_{DC}. Con dicha convertidor se alimentará el circuito electrónico y la iluminación LED.

Ilustración 4-13: convertidor AC - DC 5 V_{DC}

La otra fuente de alimentación será un convertidor AC - DC de 12 V_{DC}. Este convertidor permitirá alimentar el circuito de refrigeración, calefacción y deshumidificación.

Ilustración 4-14: convertidor AC - DC 12 V_{DC}

4.2.9. Relés

Se hará uso de relés conmutados para el control de los sistemas de refrigeración, calefacción y deshumidificación.

Cuando a dichos relés les llegue la alimentación de 5 V_{DC} procedente del circuito electrónico con su respectiva señal de datos, este conmutará y alimentará a 12 V_{DC} las conexiones correspondientes a los mencionados sistemas.

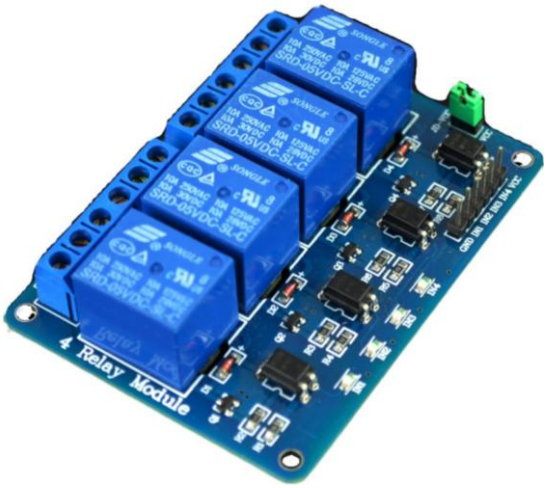

Ilustración 4-15: relés conmutados

4.2.10. Habitáculo

En una primera instancia, y como se ha ido comentando a lo largo de este proyecto, el habitáculo iba a ser diseñado y posteriormente fabricado a partir de madera de contrachapado.

Debido a que este proyecto se ha visto afectado, a fecha de marzo de 2020, por la pandemia mundial del virus Covid-19, esta parte del proyecto no ha podido realizarse. Por lo tanto, se ha tratado de buscar una alternativa lo más adecuada posible que tratase de simular el resultado final de la cámara en su conjunto.

Por ello, debido al estado de alarma y por lo tanto de confinamiento que se vive a nivel nacional, se ha podido proveer de un contenedor que ha permitido la adaptación a las necesidades de este proyecto como se muestra en la *Ilustración 4-16*.

Ilustración 4-16: contenedor - habitáculo

4.3. Sistema completo

Una vez detallado los principales componentes que formarán parte de este proyecto, se procede a la descripción del conjunto del sistema una vez ya completado.

4.3.1. Circuito electrónico

Mediante la herramienta Fritzing se han realizado los diseños finales del circuito electrónico que automatizará la cámara de cultivo.

4.3.1.1. Protoboard

En la siguiente ilustración (*Ilustración 4-17*) podemos observar la distribución en protoboard del circuito. Una ilustración más práctica y más cercana a la realidad que el esquemático adjuntado.

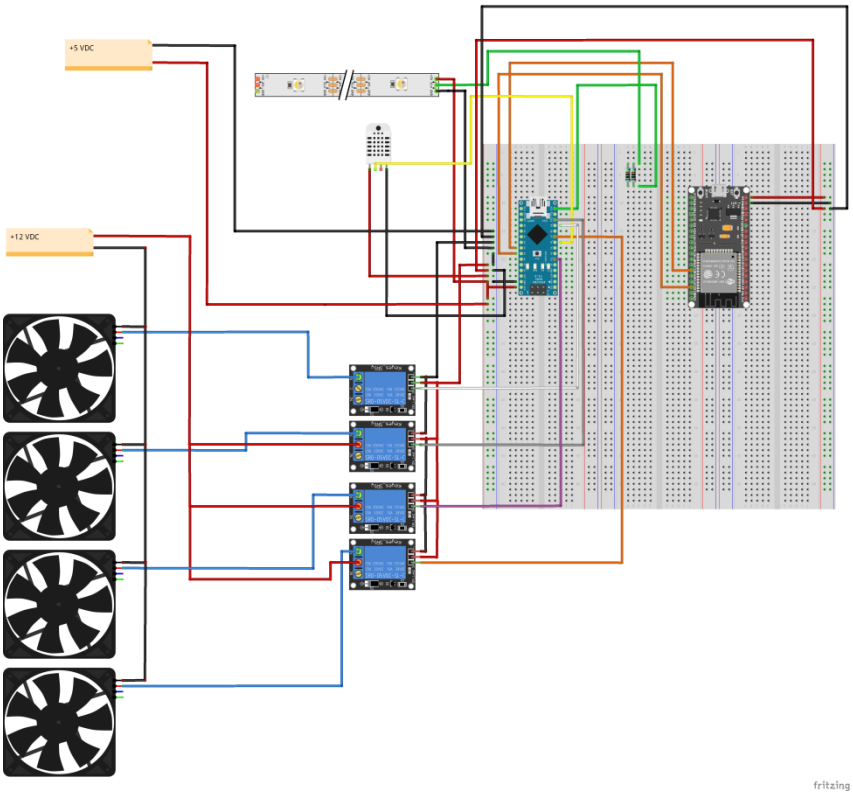

Ilustración 4-17: diseño protoboard

A continuación se analizará el funcionamiento del sistema a nivel de detalle.

El sistema se alimentará desde la red eléctrica a 230 V_{AC}. Naturalmente es requisito la utilización de convertidores a corriente continua para poder alimentar los componentes electrónicos. Se hará uso de dos convertidores mencionados con anterioridad en este capítulo.

Uno será un convertidor AC - DC de 5 V_{DC} para la alimentación del módulo ESP32, la placa Arduino Nano, el sensor de temperatura y humedad DHT22, la tira LED WS2812B y los relés conmutados.

El otro convertidor es un convertidor AC - DC de 12 V_{DC} para alimentar el sistema de refrigeración, calefacción y deshumidificación. Cuando a los relés les llegue tanto la alimentación de 5 V_{DC} como la señal de datos, estos conmutarán y alimentarán con los 12 V_{DC} a cada uno de los sistemas mencionados cuando les corresponda sujetos a la programación que se verá en el capítulo correspondiente.

Al aspecto de comunicación y conexionado del sistema se refiere, el módulo ESP32 y la placa Arduino Nano estarán comunicadas mediante el bus I^2C con la programación y cableado que conlleva. El resto de componentes siguen un cableado estándar en sus respectivos pines de señal de datos, alimentación y tierra.

Todo este sistema se podrá monitorizar a través de Bluetooth mediante una aplicación Android que se diseñará en paralelo a su construcción.

4.3.2. Montaje final

En la *Ilustración 4-18* podemos observar la distribución final y física del sistema completo.

En esta imagen, además, se ilustra la iluminación LED que recrea la iluminación (sobre todo los rayos ultravioleta) que las plantas necesitan para su germinación y crecimiento

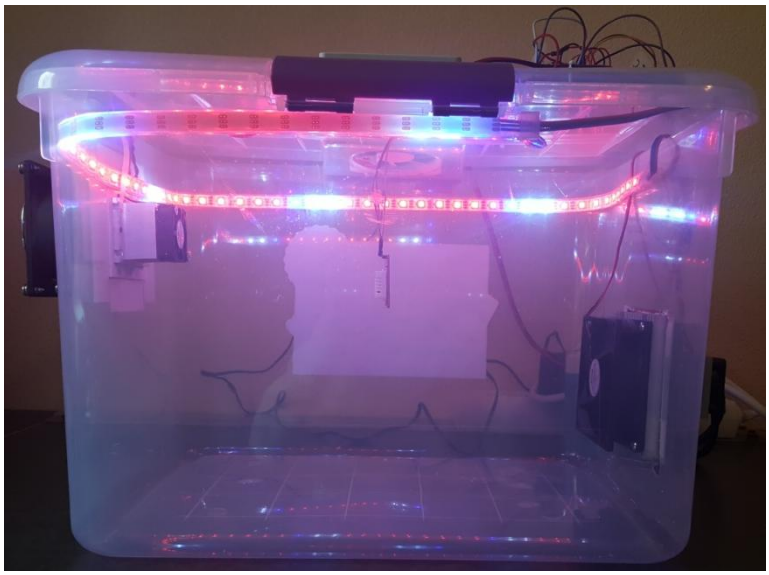

Ilustración 4-18: sistema final

A continuación se adjunta una imagen que ilustra la iluminación LED desde la perspectiva de la parte superior del contenedor.

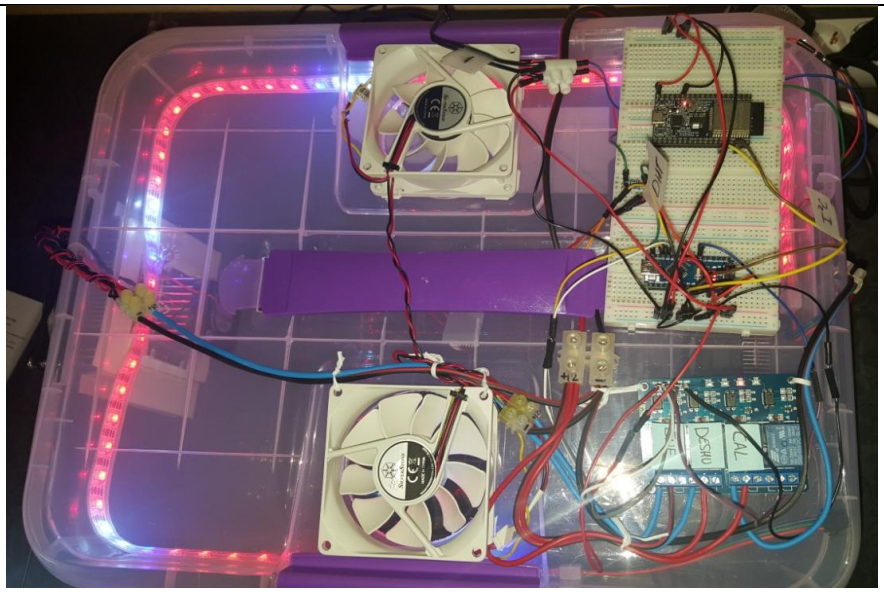

Ilustración 4-19: parte superior con iluminación LED

En la siguiente ilustración se destaca el componente situado en la parte inferior del contenedor. Se trata del sistema de calefacción mediante el cual, se introducirá aire caliente que permitirá calentar el interior cuando sea necesario. Su ubicación es debido a que se aprovechan los fundamentos de los flujos de aire.

El aire caliente es ascendente por lo que si ubicamos nuestro sistema abajo, calentará todo el interior desde abajo hasta la parte superior y por lo tanto, además de un mejor aprovechamiento de la energía de este dispositivo, no será necesario otro dispositivo de calefacción.

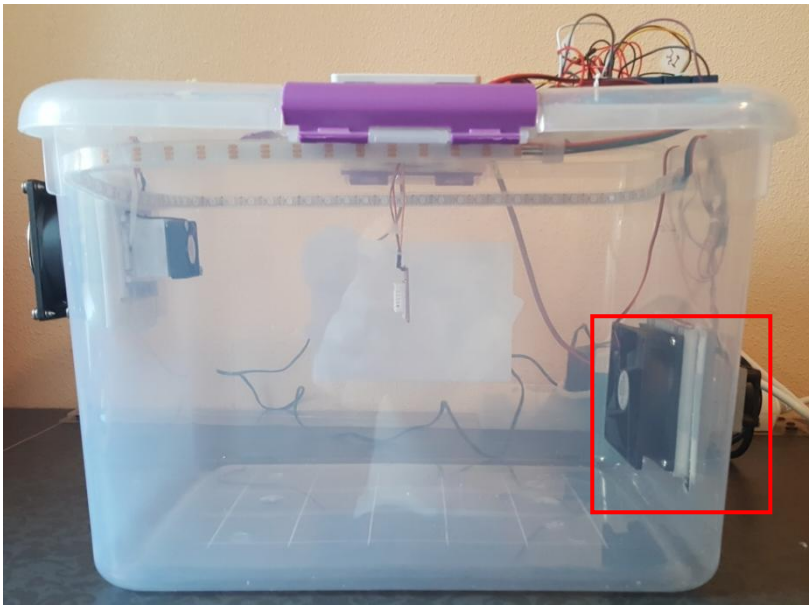

Ilustración 4-20: ubicación del sistema de calefacción

En la *Ilustración 4-21* se destaca el componente de la parte superior del contenedor. Se trata del sistema de refrigeración colocado ahí para el enfriamiento de la caja cuando sea necesario. Se ubica en ese punto debido a dos razones. Ya que el aire frío pesa más este flujo de aire frío que se introducirá en el momento de la refrigeración bajará y enfriará todo el habitáculo. Por otro lado, y precisamente por el peso del aire frío, permitirá la recirculación del aire que hay en el interior de la cámara, necesaria para el bienestar de las plantas para que el aire no se estanque.

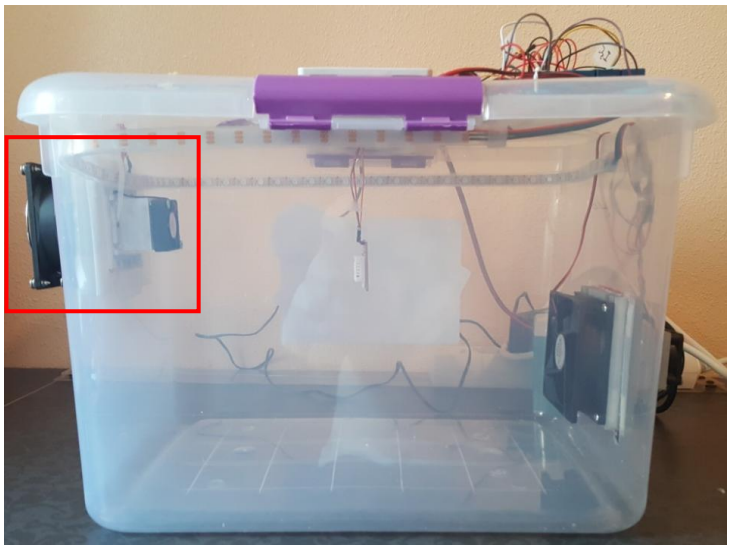

Ilustración 4-21: ubicación del sistema de refrigeración

A continuación se adjunta la *Ilustración 4-22* donde se puede observar los dos ventiladores, ambos forman el sistema de deshumidificación, y el circuito electrónico que controlará y automatizará todos los componentes que permiten mantener el clima deseado en el interior del habitáculo.

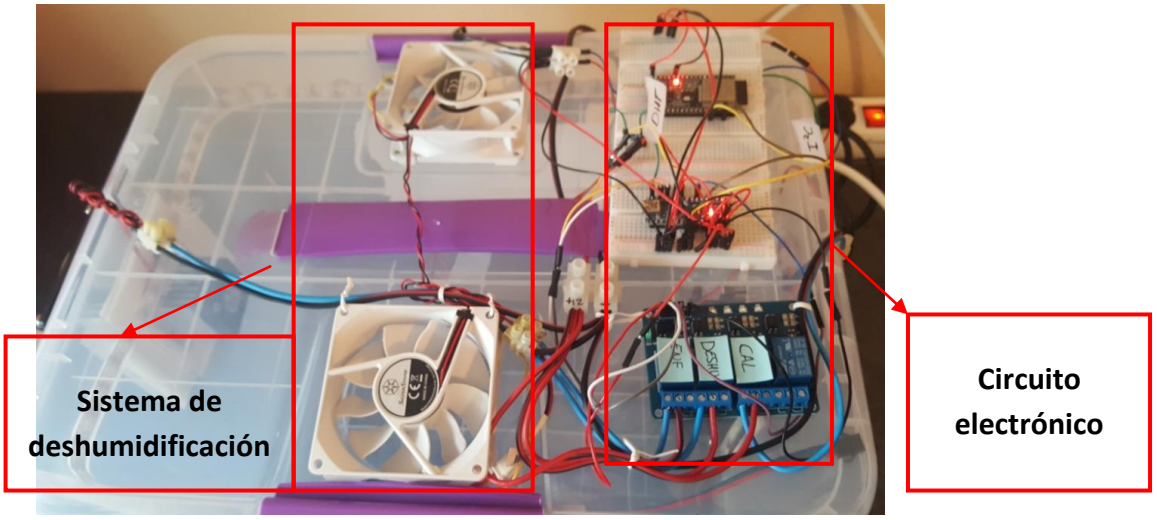

Ilustración 4-22: techo del contenedor

Debido a la situación extraordinaria de estado de alarma que se ha comentado con anterioridad, no se ha podido disponer de componentes para el sistema de humidificación pero en el software quedará programado y preparado para su uso y se indicará que funciona mediante un LED conectado al último relé disponible.

Aquí acabaría el montaje físico de la cámara de cultivo. En el siguiente capítulo se expondrá la programación realizada para el correspondiente funcionamiento.

Capítulo 5

SOFTWARE

5.1. Introducción

Una vez descrito en el capítulo anterior el hardware del sistema, en este capítulo se procederá al desglose y desarrollo de la parte del software del sistema que acompañará al hardware.

Dada la complejidad del software, debido a que se programará en tres componentes diferentes (ESP32, Arduino Nano y aplicación Android), este capítulo se irá dividiendo en diferentes apartados para mayor claridad y entendimiento de su funcionamiento.

5.2. Programación del módulo ESP32

Situado el módulo ESP32 como punto central y de control del sistema, se desarrollará a partir de éste la rama principal del software conectada a los otros dos módulos (Arduino Nano y aplicación Android). Este dispositivo se programará en el entorno de Arduino IDE con la respectiva librería adecuada para la tarjeta ESP32.

Inicialmente el ESP32 no se encontrará en modo de bajo consumo, ya que la alimentación del sistema no será a través de baterías. Este módulo estará a la espera de recibir una orden o petición desde la aplicación Android, interfaz del sistema.

Dado que la comunicación se realizará vía Bluetooth, una vez inicializado se llevará a cabo la función de comprobar el correcto funcionamiento y conexión del Bluetooth.

```
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif
```

Definimos las variables del maestro.

```
BluetoothSerial SerialBT;
const byte I2C_SLAVE_ADDR = 0X20;

float dato;
float Temp; //Temperatura ambiente medida
float Hum;  //Humedad ambiente medida

//Variable lectura BT
String comando;

//Variables control temperatura y humedad
float tempINF;  //Límite inferior Peltier
float tempSUP;  //Límite superior Peltier
float humINF;   //Límite inferior humidificación-deshumidificación
float humSUP;   //Límite superior humidificación-deshumidificación

int RequestBT;

unsigned long currentMillis;
unsigned long previousMillis = 0;
const long interval = 1000; //1000 ms
```

Una vez comprobado el funcionamiento del Bluetooth se procederá al inicio y conexión con la aplicación Android.

```
void setup() {
  Serial.begin(115200);
  SerialBT.begin("ESP32"); //Bluetooth device name
  Serial.println("Arrancamos");
  Wire.begin();
}
```

Y a la inicialización de algunas variables.

```
//Inicialización variables control temperatura y humedad
  tempINF=24; //Límite inferior Peltier
  tempSUP=26; //Límite superior Peltier
  humINF=40;  //Límite inferior humidificación-deshumidificación
  humSUP=75;  //Límite superior humidificación-deshumidificación


  RequestBT=0; //El bluetooth no está pidiendo nada
```

A continuación recibirá por Bluetooth las peticiones programadas en la aplicación.

```
void loop() {

  if (SerialBT.available()) {

    RequestBT=1;

    comando = SerialBT.readString();
    sendToSlave(comando);
    requestToSlave();
  }
```

El ESP32 está configurado y programado para ser el maestro en la comunicación vía bus I^2C con la tarjeta Arduino Nano y por lo tanto tendrá la siguiente estructura para recibir y comunicar peticiones al Arduino Nano (el cual, como veremos posteriormente, realiza el trabajo de esclavo), y entregar los datos solicitados por la aplicación.

Además, el maestro realizará una sería de órdenes programadas aparte de las recibidas por Bluetooth para el control de la temperatura y la humedad. Leerá ambas y además controlará que esta se mantenga en un intervalo adecuado de temperatura y humedad. Cuando dicho intervalo no se cumpla, según la condición, activará el sistema de refrigeración o de calefacción.

```
currentMillis = millis();
if((currentMillis - previousMillis >= interval) && (!SerialBT.available())){
  previousMillis = currentMillis;

  RequestBT=0;

  comando ="T";
  sendToSlave(comando);
  requestToSlave();
  comando = "H";
  sendToSlave(comando);
  requestToSlave();

// CONTROL DE TEMPERATURA Y HUMEDAD - ÓRDENES ENVIADAS Al ESCLAVO (Arduino Nano)
  if (Temp < tempINF && Temp > 0){
    Serial.println("Temp < tempINF");
    comando = "Z";
    sendToSlave(comando); //El esclavo (Arduino Nano) deberá activar el Peltier de calefacción
    requestToSlave();
  }
  if (Temp > tempINF){
    Serial.println("Temp > tempINF");
    comando = "X";
    sendToSlave(comando); //El esclavo (Arduino Nano) deberá apagar el Peltier de calentar
    requestToSlave();
  }
```

```
  if (Temp < tempSUP){
    Serial.println("Temp < tempSUP");
    comando = "W";
    sendToSlave(comando); //El esclavo (Arduino Nano) deberá apaga el Peltier de enfriar
    requestToSlave();
  }
  if (Hum > humSUP){
    Serial.println("Hum > humSUP");
    comando = "L";
    sendToSlave("L"); //El esclavo (Arduino Nano) deberá activar los ventiladores de deshumidificación
    requestToSlave();
}
if (Hum < humSUP && Hum > 0){
    Serial.println("Hum < humSUP");
    comando = "V";
    sendToSlave("V"); //El esclavo (Arduino Nano) deberá desactivar los ventiladores de deshumidificación
    requestToSlave();
}
if (Hum < humINF && Hum > 0){
    Serial.println("Hum < humIN");
    comando = "K";
    sendToSlave("K"); //El esclavo (Arduino Nano) deberá activar la humidificación
    requestToSlave();

if (Hum > humINF){
    Serial.println("Hum > humINF");
    comando = "R";
    sendToSlave("R"); //El esclavo (Arduino Nano) deberá desactivar la humidificación
    requestToSlave();
```

Se codifica la comunicación o transmisión de datos entre el ESP32 y el Arduino Nano mediante el bus I^2C.

```
void sendToSlave(String cmd)
{
  Wire.beginTransmission(I2C_SLAVE_ADDR);
  Wire.write((byte*)&cmd, sizeof(cmd));
  Wire.endTransmission();
}
```

Luego se desarrolla cada uno de los bloques que permitirá realizar la petición y recibir los resultados solicitados al Arduino Nano y proporcionados por este último. En este caso serán peticiones de temperatura, humedad, control de temperatura, control de humedad y control de iluminación LED.

```
void requestToSlave()
{
  dato=0;
  Wire.requestFrom(I2C_SLAVE_ADDR, sizeof(dato));

  uint8_t index = 0;
  byte* pointer = (byte*)&dato;
  while (Wire.available())
  {
    *(pointer + index) = (byte)Wire.read();
    index++;
  }
  if(comando=="T"){
    Serial.print("Temperatura: ");
    Serial.println(dato);
    SerialBT.println(dato);
  }
  else if(comando=="H"){
    Serial.print("Humedad: ");
    Serial.println(dato);
    SerialBT.println(dato);
  }
}
```

```
else if(comando=="O") {
  Serial.print("Respuesta encendido: ");
  Serial.println(dato);
}
else if(comando=="A") {
  Serial.print("Respuesta apagado: ");
  Serial.println(dato);
}
else if (comando=="M"){
  Serial.println("Límite inferior");
  Serial.println(tempINF);
  if (RequestBT == 1) SerialBT.println(tempINF);
}
else if (comando=="I"){
  Serial.println("Límite inferior +1");
  Serial.println(++tempINF);
  if (RequestBT == 1) SerialBT.println(tempINF);
}
else if (comando=="J"){
  Serial.println("Límite inferior -1");
  Serial.println(--tempINF);
  if (RequestBT == 1) SerialBT.println(tempINF);
}
else if (comando=="N"){
  Serial.println("Límite superior");
  Serial.println(tempSUP);
  if (RequestBT == 1) SerialBT.println(tempSUP);
}
```

```
else if (comando=="F"){
  Serial.println("Límite superior +1");
  Serial.println(++tempSUP);
  if (RequestBT == 1) SerialBT.println(tempSUP);
}
else if (comando=="G"){
  Serial.println("Límite superior -1");
  Serial.println(--tempSUP);
  if (RequestBT == 1) SerialBT.println(tempSUP);
}
else if (comando=="D"){
  Serial.println("Límite de deshumidifación");
  Serial.println(humSUP);
  if (RequestBT == 1) SerialBT.println(humSUP);
}
else if (comando=="Ñ"){
  Serial.println("Límite de deshumidifación +1");
  Serial.println(++humSUP);
  if (RequestBT == 1) SerialBT.println(humSUP);
}
else if (comando=="Q"){
  Serial.println("Límite de deshumidifación -1");
  Serial.println(--humSUP);
  if (RequestBT == 1) SerialBT.println(humSUP);
}
else if (comando=="U"){
  Serial.println("Límite de humidifación");
  Serial.println(humINF);
  if (RequestBT == 1) SerialBT.println(humINF);
}
```

```
else if (comando=="S"){
  Serial.println("Límite de humidifación +1");
  Serial.println(++humINF);
  if (RequestBT == 1) SerialBT.println(humINF);
}
else if (comando=="E"){
  Serial.println("Límite de humidifación -1");
  Serial.println(--humINF);
  if (RequestBT == 1) SerialBT.println(humINF);
```

Aquí acabaría el código correspondiente a la programación del módulo ESP32 (maestro).

En la siguiente sección se explicará el código para el módulo Arduino Nano (esclavo).

5.3. Programación del módulo Arduino Nano

A continuación se presentará el funcionamiento del programa del módulo Arduino Nano, programado también bajo el entorno de Arduino IDE.

En este apartado se explicará como el Arduino Nano ejecuta las peticiones recibidas del módulo ESP32 y como le envía los datos solicitados. Es decir, lectura de sensores,

envío de resultados, control de iluminación y, activación y desactivación del sistema de refrigeración, calefacción y deshumidificación cuando sea necesario.

Primero se incluyen las librerías necesario para poder trabajar con los elementos de la cámara, así como para la comunicación vía bus I^2C y la iluminación LED.

```
//CÓDIGO DEL ESCLAVO
#include "Wire.h"
#include "DHT.h"
#include <Adafruit_NeoPixel.h>
```

Después, se declaran las variables para cada bloque de operaciones que realizará Arduino Nano.

```
//VARIABLES I2C
const byte I2C_SLAVE_ADDR = 0x20;
char ordenRecibida;

//VARIABLES TIRA LED
#define PIN 12
#define NUMPIXELS 60
#define DELAYVAL 500

Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
int numBloques = 5;
int tamBloqueRojo = 9;
int tamBloqueBlanco = 2;
int tamBloqueAzul = 1;
int tamBloque = tamBloqueRojo + tamBloqueBlanco + tamBloqueAzul;
int nb,i,j,k;

int ONOFF;
```

```
//DHT22
#define DHTTYPE DHT22

const int sensorDHT = 6;
DHT dht(sensorDHT, DHTTYPE);

float temperatura; //variable global de temperatura visible en todo el código.
float humedad; //variable global de humedad visible en todo el código.

unsigned long currentMillis;
unsigned long previousMillis = 0;
const long interval = 1000; //1000 ms

//PELTIER - CONTROL TEMPERATURA
const int pinPeltierREF = 9; // pin digital D9 - relé enfriar
const int pinPeltierCAL = 10; // pin digital D10 - relé calentar
//const float thresholdLOW = 24.0;
//const float thresholdHIGH= 26.5;

//CONTROL HUMEDAD
const int pinDESHUMI = 3; //pin digital D3 - relé deshumidificar
const int pinHUMIDI = 7; //pin digital D7 - relé humidificar
```

Se programan los pines y la transmisión de datos con el módulo ESP32 vía bus I^2C.

```
void setup()
{
    Serial.begin(115200);

    pinMode(pinEnfriar, OUTPUT);
    pinMode(pinCalentar, OUTPUT);
    dht.begin();

    Wire.begin(I2C_SLAVE_ADDR);
    Wire.onReceive(receiveEvent);
    Wire.onRequest(requestEvent);
}
```

Se declaran los bloques principales para recibir peticiones y para cumplir dichas peticiones y enviar los resultados solicitados. Además, se declaran e inicializan funciones para la lectura de la temperatura y humedad para el control de estas últimas, así como para el funcionamiento de la iluminación LED.

```
void setup() {
  Serial.begin(9600);
  Serial.println("Arrancamos Nano");
  Serial.println("DHTXX test");
  dht.begin();

  Wire.begin(I2C_SLAVE_ADDR);
  Wire.onReceive(receiveEvent);
  Wire.onRequest(requestEvent);

  //Lectura inicial al arrancar el programa
  temperatura = dht.readTemperature();
  humedad = dht.readHumidity();

  //Peltier
  pinMode(pinPeltierREF, OUTPUT);  //definir pin como salida
  pinMode(pinPeltierCAL, OUTPUT);  //definir pin como salida

  //Control humedad
  pinMode(pinDESHUMI, OUTPUT); //definir pin como salida
  pinMode(pinHUMIDI, OUTPUT); //definir pin como salida

  //LED
  pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
  apagarTiraLED();

  ONOFF = 0;
}
```

```
void receiveEvent(int bytes)
{
   uint8_t index = 0;
   while (Wire.available())
   {
      byte* pointer = (byte*)&ordenRecibida;
      *(pointer + index) = (byte)Wire.read();
      index++;
   }

}
```

Lo siguiente es programar al Arduino Nano para que reciba las órdenes del maestro por el bus I^2C. Según la orden recibida realizará un acción o llamará a una función a ejecutar.

```
void requestEvent()
{
 Serial.println(ordenRecibida);

   if(ordenRecibida == 'T') {
      Serial.println("Envio temperatura");
      Wire.write((byte*)&temperatura, sizeof(temperatura));
   }
   else if(ordenRecibida == 'H') {
      Serial.println("Envio humedad");
      Wire.write((byte*)&humedad, sizeof(humedad));
   }
   else if(ordenRecibida == 'O') {
      Serial.println("Encender");
      ONOFF = 1;
   }
    else if(ordenRecibida == 'A') {
      Serial.println("aPAGAR");
      ONOFF = 0;
   }
   else if (ordenRecibida == 'Y'){
      Serial.println("Refrigerar");
      digitalWrite(pinPeltierREF, LOW);   // Encender la placa Peltier para enfriar
   }
   else if (ordenRecibida == 'Z'){
      Serial.println("Calentar");
      digitalWrite(pinPeltierCAL, LOW);   // Encender la placa Peltier para calentar
   }
```

```
else if (ordenRecibida == 'X'){
   Serial.println("No Calentar");
   digitalWrite(pinPeltierCAL, HIGH);   // Apagar la placa Peltier para calentar
}
else if (ordenRecibida == 'W'){
 Serial.println("No enfriar");
 digitalWrite(pinPeltierREF, HIGH);   // Apagar la placa Peltier para enfriar
}
else if (ordenRecibida == 'L'){
 Serial.println("Deshumidificar");
 digitalWrite(pinDESHUMI, LOW);   // Activar deshumidificación
}
else if (ordenRecibida == 'V'){
 Serial.println("No Deshumidificar");
 digitalWrite(pinDESHUMI, HIGH);   // Desactivar deshumidificación
}
else if (ordenRecibida == 'K'){
 Serial.println("Humidificar");
 digitalWrite(pinHUMIDI, LOW);   // Activar humidificación
}
else if (ordenRecibida == 'R'){
 Serial.println("No Humidificar");
 digitalWrite(pinHUMIDI, HIGH);   // Desactivar humidificación
}
```

Aunque en el montaje físico no se ha podido disponer de los medios para obtener e implementar un componente de humidificación debido al estado de alarma y de confinamiento que se está viviendo actualmente en el país (esto ha sido ya comentado en el capítulo anterior), se ha querido dejar preparado y programado el bloque de dicho sistema para una posible mejora en trabajos futuros. Este sistema corresponde a las dos últimas órdenes llamadas "K" y "R".

Como se expone en el Capítulo 4 se indicará el funcionamiento del sistema mediante un LED conectado a los relés.

A continuación se programa el bloque de lectura de temperatura y humedad, así como el control del encendido y el apagado de la tira LED.

```
void loop() {

// TEMPERATURA Y HUMEDAD - DHT
  currentMillis = millis();
  if(currentMillis - previousMillis >= interval){
    previousMillis = currentMillis;

    temperatura = dht.readTemperature();
    humedad = dht.readHumidity();

// LED
   if (ONOFF == 1){
    encenderTiraLED();
   }
   else apagarTiraLED();

}
}
```

Y por último, las funciones que se encargarán de ejecutar el encendido y apagado de la iluminación LED así como de la secuencia de colores de la misma.

```
void encenderTiraLED() {

    for(int nb = 0; nb < numBloques; nb++) {

    //Bloque de rojo
    for(i = nb*tamBloque; i < (nb*tamBloque)+tamBloqueRojo ; i++){
      pixels.setPixelColor(i, pixels.Color(96,0,0));
    }

    //Bloque de blanco
    for(j = i; j < i+tamBloqueBlanco; j++) {
      pixels.setPixelColor(j, pixels.Color(96,96,96));
    }

    //Bloque de azul
    for(k = j; k < j+tamBloqueAzul; k++) {
      pixels.setPixelColor(k, pixels.Color(0,0,96));
    }
    pixels.show();
    pixels.show();

  }
}

void apagarTiraLED() {
  pixels.clear();
  pixels.show();
  pixels.show();
}
```

5.4. Comunicación ESP32 - Arduino Nano

Como se ha comentado en secciones anteriores, se va a comunicar el dispositivo ESP32 con el dispositivo Arduino Nano mediante el bus I^2C.

Bus I^2C

El interés por el bus I^2C existe porque, de forma similar a lo que pasaba con el bus SPI, una gran cantidad de dispositivos disponen conexión mediante I^2C, como acelerómetros, brújulas, displays, etc.

Para su funcionamiento sólo se requieren dos cables, uno para la señal de reloj (CLK) y otro para el envío de datos (SDA), lo cual es una ventaja frente al bus SPI. Por contra, su funcionamiento es un poco más complejo, así como la electrónica necesaria para implementarla.

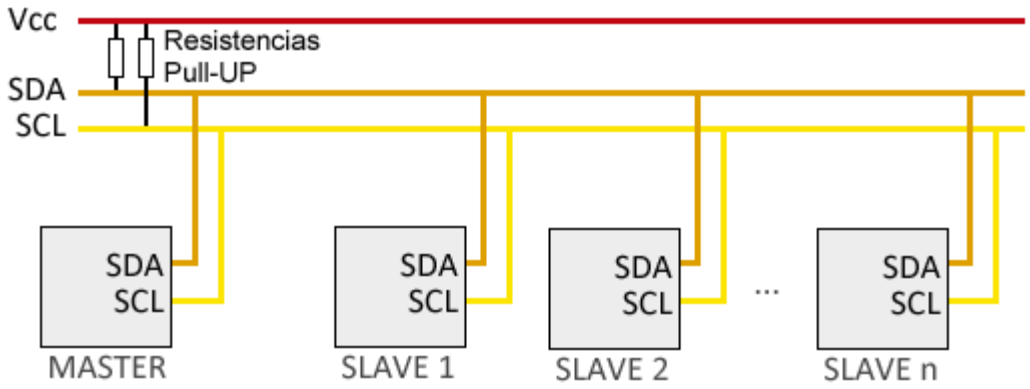

Ilustración 5-1: Esquema de conexión del bus I^2C

En este bus cada dispositivo dispone de una dirección que se emplea para acceder a los dispositivo de forma individual. Esta dirección se puede fijar mediante (en cuyo caso, frecuentemente, se pueden modificar los últimos 3 bits mediante *jumpers* o interruptores) o totalmente por software.

En general, cada dispositivo conectado al bus debe tener una dirección única. Si tenemos varios dispositivos similares tendremos que cambiar la dirección o, en caso de no ser posible, implementar un bus secundario.

Como se ha desarrollado en secciones anteriores, el bus I^2C tiene una arquitectura de tipo maestro-esclavo. El dispositivo maestro inicia la comunicación con los esclavos, y puede mandar o recibir datos de los esclavos. Los esclavos no pueden iniciar la comunicación (el maestro tiene que preguntarles), ni hablar entre sí directamente. Es posible disponer de más de un maestro, pero sólo uno puede ser el maestro cada vez. Es decir, permite un protocolo de comunicación *multi-master* y *multi-slave*.

Se recuerda que el bus I^2C es síncrono y, por lo tanto, el maestro proporcionará una señal de reloj que mantendrá sincronizados a todos los dispositivos del bus.

Otro aspecto importante del bus I^2C es que su protocolo prevé resistencias *Pull-Up* en las líneas a Vcc. Arduino ya dispone de resistencias internas *Pull-Up* que se pueden activar con la librería *Wire*. Sin embargo estas resistencias tienen un valor de entre 20 - 30 kΩ y, por lo tanto, son resistencias muy bajas. Esto implicará que los flancos de subida de la señal serán menos rápidos, por lo que las velocidades que se pueden usar tienen que ser velocidades bajas y además distancias de comunicación menores. Si se quieren emplear velocidades o distancias de transmisión superiores, se deberán poner, físicamente, resistencias de *Pull-Up* (aconsejablemente entre 1 kΩ a 4K7). En esta ocasión, no será necesario implementar más resistencia *Pull-Up*.

Comunicación I²C entre ESP32 y Arduino Nano

El bus I²C no sirve sólo para comunicarnos con todo tipo de sensores, también podemos usarlo para conectar dos o más microprocesadores usando únicamente dos cables para la comunicación.

Esquema de conexión.

La conexión física no es compleja. Como se ha comentado anteriormente, el bus I²C utiliza dos cables. Uno es utilizado para la señal del reloj (SCL) y el otro para enviar y recibir datos (SDA). Por lo tanto, se alimenta la alimentación (3,3-5 V y GND) de ambos Arduinos (ESP32 y Arduino Nano), y los dos pines del I²C (SDA, SCL) de un Arduino (ESP32) a los pines SDA y SCL del otro (Arduino Nano).

I²C	ESP32	Arduino Nano
SDA	SDA (GPIO21)	SDA (A4)
SCL	SCL (GPIO22)	SDA (A5)
GND	GND	GND
VCC	3,3 - 5 V	3,3 - 5 V

Tabla 5-1: Esquema de conexiones del bus I²C - ESP32 - Arduino Nano

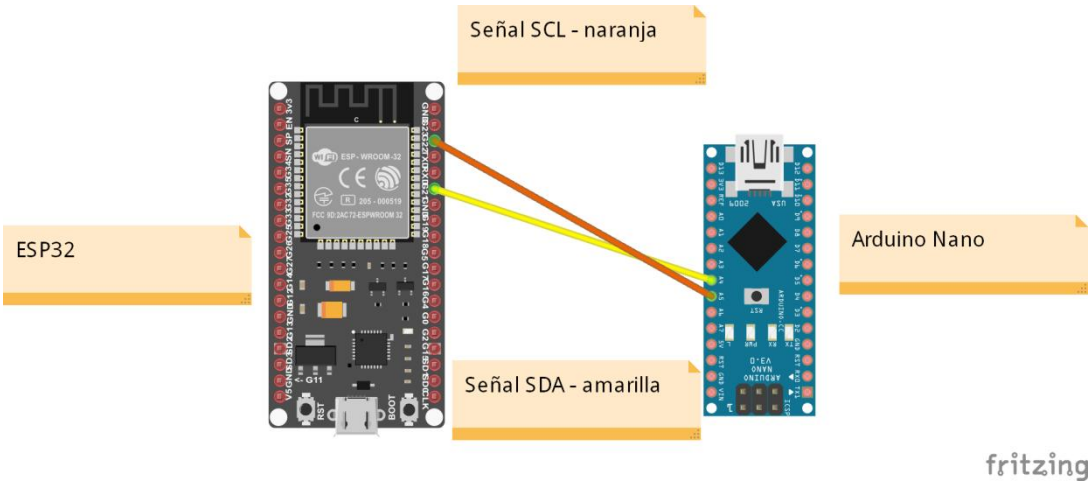

Ilustración 5-2: Pinout bus I²C en ESP32 - Arduino Nano

5.5. Aplicación Android

Para finalizar en este capítulo se desglosará como se ha diseñado, creado y programado la aplicación que hará de interfaz con el usuario y permitirá monitorizar y controlar el sistema de la cámara de cultivo.

Para su desarrollo se ha utilizado la herramienta online para Android MIT App Inventor.

El primer paso para su creación es el diseño de la pantalla con la que vamos a interactuar, que componentes van a aparecer y como van a estar distribuidos. La aplicación para este proyecto va a presentar la siguiente pantalla de inicio, ilustrada en la siguiente imagen.

Ilustración 5-3: pantalla inicial App Inventor

En la *ilustración 5-4* vemos ilustrada el aspecto del menú desplegable.

Ilustración 5-4: menú desplegable App Inventor

Una vez que hemos diseñado la aplicación queda programarla para su uso. Su lenguaje de programación es de conexión entre bloques. En las siguientes ilustraciones se muestra la programación de la aplicación diseñada para este proyecto.

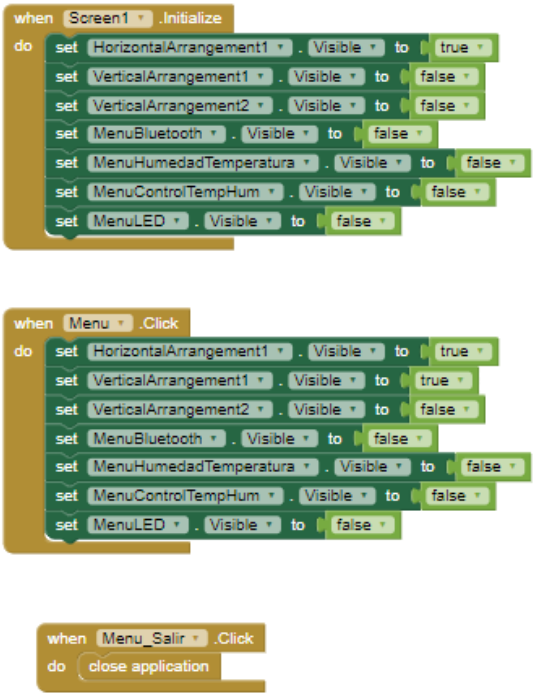

Ilustración 5-5: código de bloques - parte 1

En la parte de código mostrada en la imagen anterior (*Ilustración 5-5*) queda reflejado la programación en lenguaje de bloques correspondiente al menú desplegable de la aplicación así como la programación del botón *Salir* de la aplicación que permitirá cerrarla.

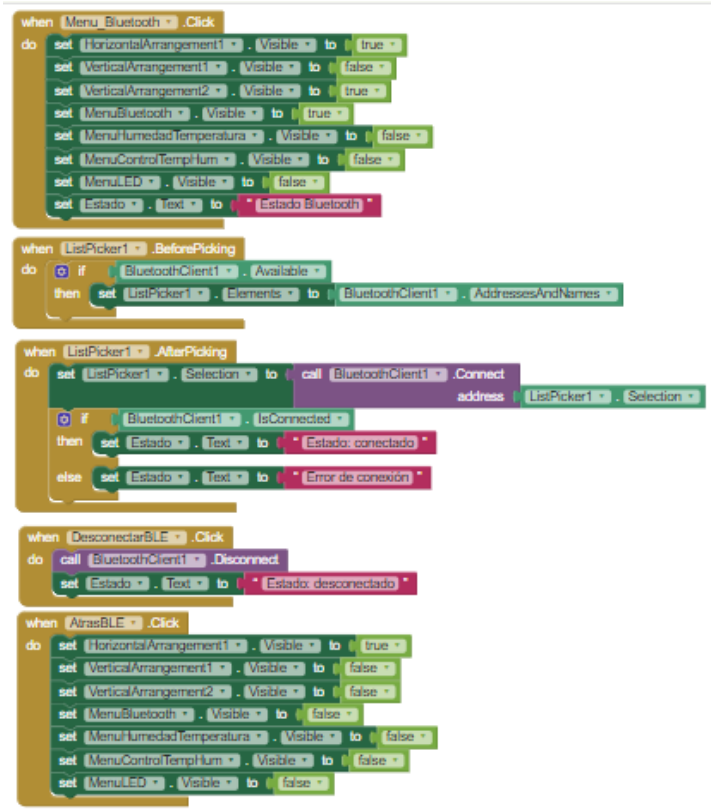

Ilustración 5-6: código de bloques - parte 2

En la *Ilustración 5-6* se puede observar el bloque de programación correspondiente a la configuración de Bluetooth donde se conectará, buscará los dispositivos disponibles y permitirá conectarse al dispositivo que se seleccione. Además se configura el botón *Desconectar* para desconectar el Bluetooth así como el botón *Atrás* para volver a la pantalla inicial de la aplicación.

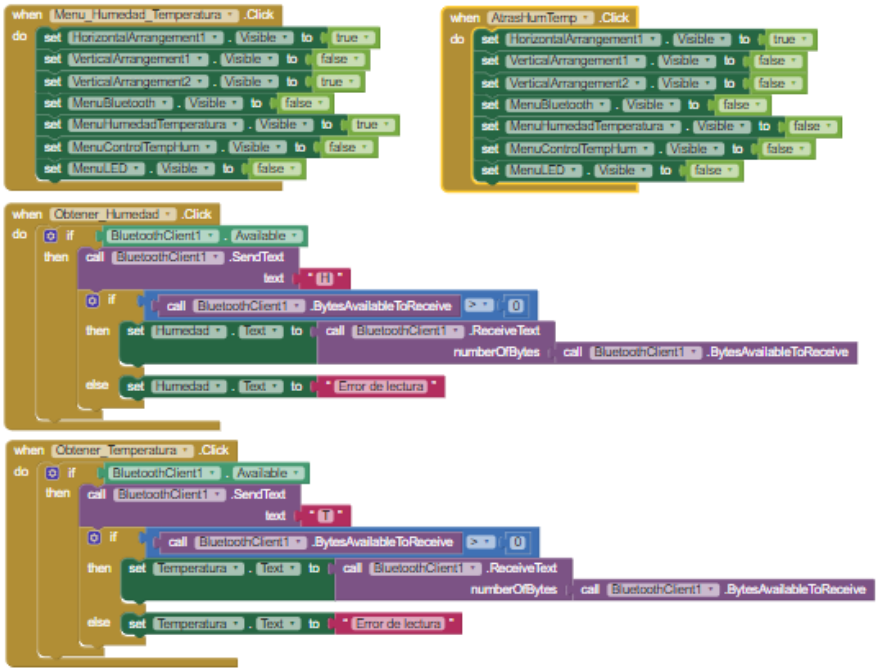

Ilustración 5-7: código de bloques - parte 3

En la parte tres del código (*Ilustración 5-7*), se presenta la programación para el control del menú correspondiente a la temperatura. Se programa la solicitud de temperatura así como el botón *Atrás* para volver a la pantalla inicial.

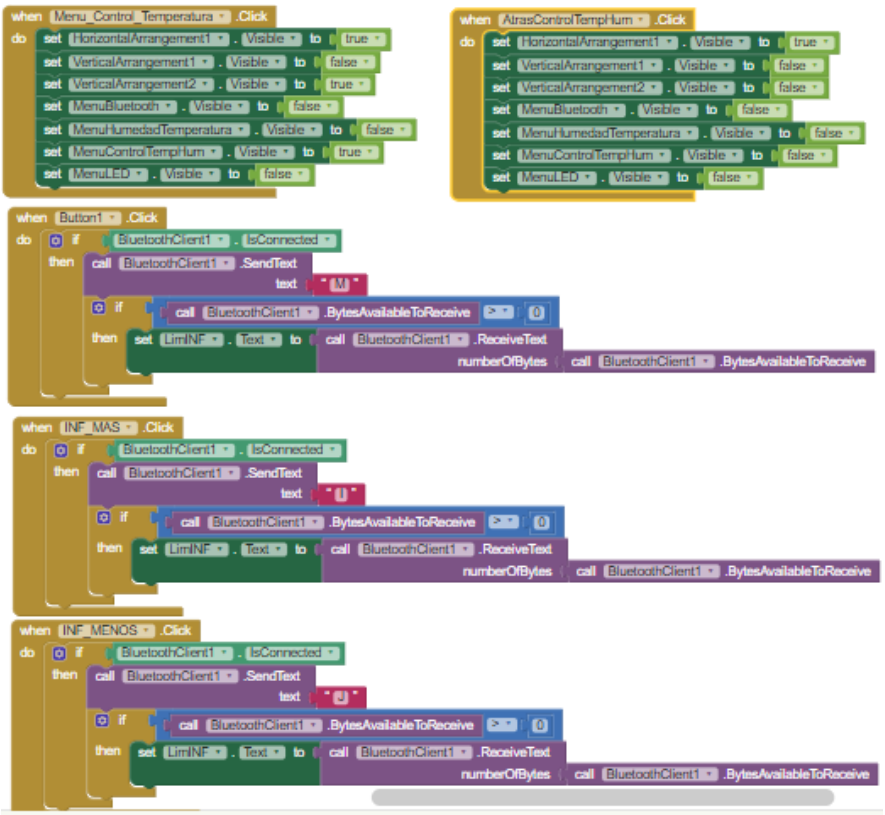

Ilustración 5-8: código de bloques - parte 4

En la *Ilustración 5-8* se refleja la configuración para el control de los límites de temperatura. En este caso, permite modificar el límite inferior no se quiera bajar. Como en bloques anteriores, se configura el botón de *Atrás* para poder retroceder.

Ilustración 5-9: código de bloques - parte 5

En esta parte del código (*Ilustración 5-9*), se programa la parte de control donde se puede modificar el límite superior de temperatura que no se quiere sobrepasar.

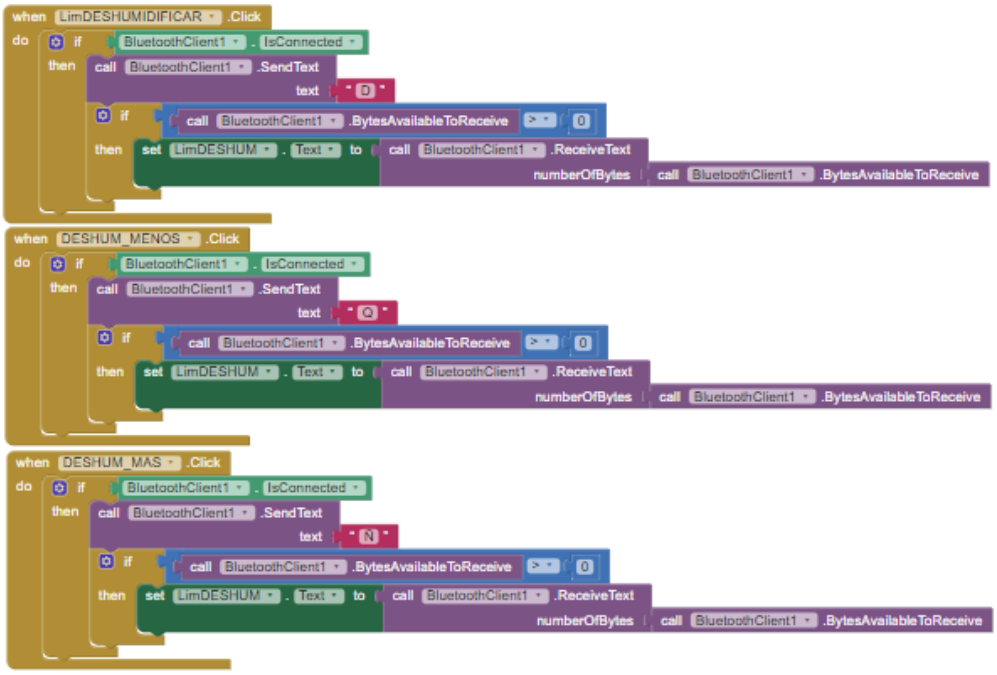

Ilustración 5-10: código de bloques - parte 6

En esta sección de código (*Ilustración 5-10*) se programa el bloque que modificará si queremos, al igual que con la temperatura, el límite de deshumidificación.

Ilustración 5-11: código de bloques - parte 7

Al igual que en el bloque anterior, en esta imagen (Ilustración 5-11) se representa la programación para la modificación de los límites de humidificación.

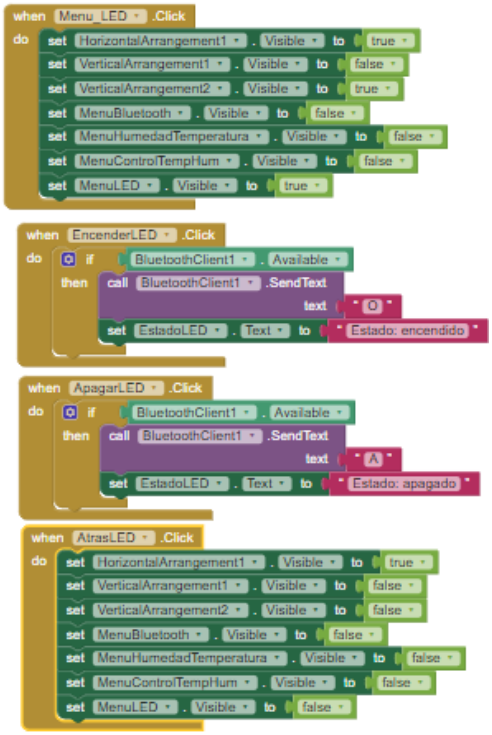

Ilustración 5-12: código de bloques - parte 8

Y por último, el código realizado mediante programación de bloques termina con el último bloque (*Ilustración 5-12*) encargado de encender y apagar la iluminación LED. Como en los bloques anteriores, también se programa un botón de *Atrás* para retroceder a la pantalla inicial.

Capítulo 6

CONCLUSIONES Y TRABAJOS FUTUROS

6.1. Conclusiones

En este proyecto se ha abordado la problemática de la existencia de cámaras de cultivo de elevado coste, así como el estudio y el desarrollo de una cámara de cultivo de bajo coste totalmente funcional como alternativa a las ya existentes que podría abrir un campo de oportunidades de cultivo para diversos ámbitos y para diversos perfiles de usuarios.

Se ha comenzado con un estudio del mercado actual y de las características fundamentales que toda cámara de cultivo debe cumplir. Después se ha procedido a un estudio de las diversas alternativas en cada aspecto y componente fundamental del caso de estudio, así como la elección final más apropiada para este proyecto teniendo en cuenta las características de uso, la implementación y el ámbito económico.

Una vez elegidos cada uno de los componentes y aspectos que iban a ser integrados a la cámara, se ha llevado a cabo un diseño completo tanto en estructura hardware como en estructura software del objeto de estudio. Usando el entorno de Arduino se han implementado las librerías necesarias y se ha desarrollado el programa que permitirá el monitoreo y la automatización de la cámara de cultivo.

Una vez todo preparado se ha procedido a la puesta en marcha, con éxito, de la cámara de cultivo con todos los componentes implementados y una monitorización realizada a través de la aplicación Android también diseñada para este proyecto.

Con este proyecto se ha querido demostrar que se pueden realizar cámaras de cultivo económicas e innovadoras con gran capacidad de evolución.

En el ámbito más personal cabe destacar que a través del estudio y desarrollo de este proyecto se ha obtenido también un aprendizaje bastante completo en el campo de la electrónica. Dicho aprendizaje ha permitido profundizar de forma práctica en aspectos ya contemplados en el grado, así como la adquisición de otros conocimientos que complementan dichos aspectos. El punto más destacable en este aprendizaje sería la variedad de usos y aplicaciones que otorgan los microcontroladores así como su programación.

Finalmente, otro aspecto fundamental que resaltar es el hecho de haber estado implicado en un caso real de diseño y automatización de un sistema electrónico con una aplicación real en la actualidad.

6.2. Trabajos futuros

El desarrollo del caso de estudio ha permitido destacar limitaciones que desencadenaban en la posibilidad de mejoras y de posibles sistemas que podrían agregarse complementando y perfeccionando así el proyecto. Esto aumentaría su capacidad de aplicación y uso convirtiéndolo en una herramienta más real y más práctica.

Uno de los posibles trabajos futuros que se ha contemplado es una extensión de dicho proyecto llevándolo a un nivel por encima del actual. El diseño y creación de una cámara de mayor volumen adaptada para acoger diversos módulos independientes como el desarrollado en este proyecto.

Esto permitiría tener varios cultivos diferentes cada uno con su ambiente climático necesario según también en qué momento de crecimiento se encuentre. Este desarrollo daría pie a una mayor versatilidad de la cámara y adaptabilidad. Se podría desarrollar perfectamente siguiendo la línea de trabajo de este proyecto pero otorgándole el tiempo necesario que requeriría.

Otro trabajo futuro, que puede ser complementario al citado anteriormente, es incluir más instrumentos y variables de medida, y añadirlas en el control de la aplicación. Como última propuesta de trabajo futura, también complementaria a las mencionadas con anterioridad, es incluir un mejor y más amplio desarrollo de la aplicación así como

de su interfaz. Esta propuesta es debida a las limitaciones detectadas en la herramienta utilizada a lo largo del proyecto. Es una herramienta válida para una aplicación más simplificada pero a la hora de un desarrollo más complejo, con mayor número de funciones y necesidades, la herramienta queda limitada provocando posibles fallos en el funcionamiento y/o soporte de la herramienta en sí, ajeno a la programación de la misma.

Cabe mencionar, como se ha hecho en capítulos anteriores, que el estado de alarma que se vive a nivel nacional provocada por la pandemia mundial del Covid-19 ha influido en este proyecto forzando a que el acabado final no sea exactamente como se programó en un principio, aunque si cabe resalta que el resultado final es muy aproximado y cumple con los objetivos principales de este proyecto.

Capítulo 7

REFERENCIAS BIBLIOGRÁFICAS

CÁMARAS DE CULTIVO:

[Cuben] Cámaras para cultivo de plantas. Disponible on-line en: https://www.cuben.com.ar/catalogos/22_CAMARA_CULTIVOS.pdf

[Mpcontrol] Cámaras climáticas. Disponible on-line en: http://www.mpcontrol.es/index.php/definicion-camara-climatica/

[Binder] Cámaras de cultivos comerciales Binder. Disponible on-line en: https://www.binder-world.com/en/products/growth-chambers

[Fishersci] Lista de precios y productos de cámaras comerciales Binder. Disponible on-line en: https://www.fishersci.es/es/es/promotions/buy-binder-growth-chamber-at-special-price.html

[EdenCPs] Cámara DIY de EdenCPs. Disponible on-line en: https://edencps.com/highland-nepenthes-chamber/

[Instructables] Cámara DIY de un usuario del blog Instructables. Disponible on-line en: https://www.instructables.com/id/DIY-Grow-Box/

HABITÁCULO:

[Blog expositores] ¿Qué es el metacrilato? ¿De qué está hecho? Disponible on-line en: https://blog.expositores-metacrilato.es/que-es-el-metacrilato/

[Faberplast] Usos del metacrilato. Disponible on-line en: https://www.faberplast.net/blog/usos-del-metacrilato/

[Pixartprinting] ¿Qué el metacrilato? La alternativa al vidrio. Disponible on-line en: https://www.pixartprinting.es/blog/que-es-metacrilato/

[Wikipedia] ¿Qué es el contrachapado? Disponible on-line en: https://es.wikipedia.org/wiki/Contrachapado

[Emedec] Contrachapado: características y aplicaciones. Disponible on-line en: https://www.emedec.com/contrachapado-caracteristicas-aplicaciones/

[Digfineart] Desventajas del uso del contrachapado. Disponible on-line en: https://www.digfineart.com/Az2N5J4zy/

[Wikipedia] ¿Qué es la formica? Disponible on-line en: https://es.wikipedia.org/wiki/Formica_(pl%C3%A1stico)

[Formica mexico] ¿Para qué se usa el laminado plástico de formica? Disponible on-line en: https://formicamexico.wordpress.com/2011/12/26/para-que-se-usa-el-laminado-plastico-de-formica/

[Termiserprotecciones] Formica: el material y sus características principales. Disponible on-line en: http://termiserprotecciones.com/formica-material/

AISLAMIENTO:

[Over-blog] Tipos de aislantes térmicos, ventajas y desventajas. Disponible on-line en: https://es.over-blog.com/Tipos_de_aislantes_termicos_ventajas_y_desventajas-1228321783-art290079.html

[Rtarquitectura] Aislamiento térmico: tipos y características. Disponible on-line en: https://www.rtarquitectura.com/aislamiento-termico-tipos-y-caracteristicas/

[Ecogreenhome] Ventajas y beneficios de los aislantes térmicos. Disponible on-line en: https://ecogreenhome.es/ventajas-beneficios-del-aislamiento-termico/

ILUMINACIÓN LED:

[Greenice] Iluminación LED para cultivo de interior. Disponible on-line en: https://greenice.com/es/blog/iluminacion-led-para-cultivo-interior-una-poderosa-herramienta--n60

[Greenrastashop] Iluminación LED. Ventajas e inconvenientes Disponible on-line en: https://www.greenrastashop.com/post/cultivo-con-leds-ventajas-e-inconvenientes

CONTROL DE TEMPERATURA:

[Fayerwayer] Tipos de sistemas de refrigeración. Disponible on-line en: https://www.fayerwayer.com/2007/03/distintos-tipos-de-refrigeracion/

[Planeta huerto] Calefactor eléctrico: ventajas y desventajas. Disponible on-line en: https://www.planetahuerto.es/revista/ventajas-e-inconvenientes-de-cada-tipo-de-calefaccion_00423

CONTROL DE HUMEDAD RELATIVA:

[Soler palau] Deshumidificar o ventilar. Disponible on-line en: https://www.solerpalau.com/es-es/blog/deshumidificador-o-ventilar-forma-eficiente/

SENSOR DE TEMPERATURA:

[Siberzone] ¿Qué es un sensor de temperatura y para qué se utiliza? Disponible on-line en: https://www.siberzone.es/blog-sistemas-ventilacion/que-es-un-sensor-de-temperatura-y-para-que-se-utiliza/

[Medir temperatura] Sensor de temperatura. Disponible on-line en: http://medirtemperatura.com/sensor-temperatura.php

[Ingmecafenix] Termistor sensor de temperatura. Disponible on-line en: https://www.ingmecafenix.com/automatizacion/termistor-sensor-temperatura/

[tr3sdland] NTC. Disponible on-line en: https://www.tr3sdland.com/2011/12/componentes-el-sensor-ntc/

[Ecured] PTC. Disponible on-line en: https://www.ecured.cu/Termistores_PTC

[Ingmecafenix] Sensor de temperatura RTD. Disponible on-line en: https://www.ingmecafenix.com/automatizacion/sensor-temperatura-rtd/

[Jmi] Termopar. Disponible on-line en: https://www.jmi.com.mx/literatura/blog/item/40-que-son-y-para-que-sirven-los-termopares.html

[Ingmecafenix] Sensor de temperatura termopar. Disponible on-line en: https://www.ingmecafenix.com/automatizacion/sensor-temperatura-termopar/

[Wikipedia] Termopar. Disponible on-line en: https://es.wikipedia.org/wiki/Termopar

SENSOR DE HUMEDAD RELATIVA:

[Ecured] Como funcionan los sensores de humedad. Disponible on-line en: https://www.ecured.cu/Sensor_de_Humedad

[Mecatronicalatam] ¿Qué es un sensor de humedad? Disponible on-line en: https://www.mecatronicalatam.com/tutorial/es/sensores/sensor-de-humedad

[Wikipedia] Tipos de sensores de humedad. Disponible on-line en: https://es.wikipedia.org/wiki/Sensor_de_humedad

SENSOR DE TEMPERATURA Y HUMEDAD RELATIVA:

[Naylamp mechatronics] Sensor de temperatura y humedad relativa DHT22 (AM2302). Disponible on-line en: https://naylampmechatronics.com/sensores-temperatura-y-humedad/58-sensor-de-temperatura-y-humedad-relativa-dht22-am2302.html

[Panamahitek] Sensores DHTXX. Disponible on-line en: http://panamahitek.com/sensores-dhtxx/

[Aosong] Hoja de características DHT22. Disponible on-line en: http://akizukidenshi.com/download/ds/aosong/AM2302.pdf

[Omniblug] Sensor de temperatura y humedad DHT11 - DHT22. Disponible on-line en: http://www.omniblug.com/sensor-temperatura-humedad-DHT11-DHT22.html

[Hwlibre] DHT22: el sensor de temperatura y humedad de precisión. Disponible on-line en: https://www.hwlibre.com/dht22/

[Luis Llamas] Medir temperatura y humedad con arduino y sensor DHT11 - DHT22. Disponible on-line en: https://www.luisllamas.es/arduino-dht11-dht22/

Arduino Nano:

[DescubreArduino.com] Arduino Nano, qué es, Pinout y características. Disponible on-line en: https://descubrearduino.com/arduino-nano-pinout/

[Aprendiendo Arduino] Arduino Nano. Disponible on-line en: https://aprendiendoarduino.wordpress.com/tag/arduino-nano/

[Electronilab] Arduino Nano V3. Disponible on-line en: https://electronilab.co/tienda/arduino-nano-v3-atmega328-5v-cable-usb/

ESP32:

[Luis Llamas] ¿Qué es el ESP32? Disponible on-line en: https://www.luisllamas.es/esp32/

[Wikipedia] El ESP32. Disponible on-line en: https://es.wikipedia.org/wiki/ESP32

[Prometec] Instalando el ESP32. Disponible on-line en: https://www.prometec.net/instalando-esp32/

[Aprendiendo Arduino] ESP32. Disponible on-line en: https://aprendiendoarduino.wordpress.com/tag/esp32/

IDE de Arduino:

[Arduino] Sitio web Arduino. Disponible on-line en: https://www.arduino.cc/en/Main/Software

[Wikipedia] IDE de Arduino. Disponible on-line en: https://es.wikipedia.org/wiki/Arduino_IDE

[Tuelectronica] ¿Qué es Arduino? Disponible on-line en: https://tuelectronica.es/que-es-arduino-ide/

MicroPython:

[Wikipedia] MicroPython. Disponible on-line en: https://es.wikipedia.org/wiki/MicroPython

RTOS:

[Wikipedia] RTOS. Disponible on-line en: https://es.wikipedia.org/wiki/Sistema_operativo_de_tiempo_real

[Guillehg] Introducción a los RTOS. Disponible on-line en: http://www.guillehg.com/index.php?option=com_content&view=article&id=49&Itemid=691

Mongoose OS:

[Wikipedia] Mongoose OS. Disponible on-line en: https://es.wikipedia.org/wiki/Mongoose_OS

[Pdacontroles] Introducción a plataforma Mongoose OS. Disponible on-line en: http://pdacontroles.com/introduccion-plataforma-mongoose-os-esp8266/

Espruino:

[Wikipedia] ¿Qué es Espruino? Disponible on-line en: https://en.wikipedia.org/wiki/Espruino

SPI:

[Panamahitek] ¿Qué es y cómo funciona el protocolo SPI? Disponible on-line en: http://panamahitek.com/como-funciona-el-protocolo-spi/

[Wikipedia] Protocolo SPI. Disponible on-line en: https://es.wikipedia.org/wiki/Serial_Peripheral_Interface

[Aprendiendo Arduino] Bus SPI. Disponible on-line en: https://aprendiendoarduino.wordpress.com/category/bus-spi/

I^2C:

[Teslabem] Fundamentos del protocolo I^2C. Disponible on-line en: https://teslabem.com/nivel-intermedio/fundamentos-del-protocolo-i2c-aprende/

[Luis Llamas] I^2C. Disponible on-line en: https://www.luisllamas.es/arduino-i2c/

[Luis Llamas] Como conectar dos Arduino por bus I^2C. Disponible on-line en: https://www.luisllamas.es/como-conectar-dos-arduino-por-bus-i2c/

[Wikipedia] I^2C. Disponible on-line en: https://es.wikipedia.org/wiki/I%C2%B2C

UART:

[Wikipedia] UART. Disponible on-line en: https://es.wikipedia.org/wiki/Universal_Asynchronous_Receiver-Transmitter

[Rinconingenieril] La UART. Disponible on-line en: https://www.rinconingenieril.es/funciona-puerto-serie-la-uart/

Host SD:

[Protocolo] Protocolo de configuración dinámica de Host. Disponible on-line en: https://es.wikipedia.org/wiki/Protocolo_de_configuraci%C3%B3n_din%C3%A1mica_de_host

[Docs.citrix] Protocolo de configuración Host. Disponible on-line en: https://docs.citrix.com/es-es/citrix-sd-wan-orchestrator/site-level-configuration/dhcp.html

Android:

[Wikipedia] Android. Disponible on-line en: https://es.wikipedia.org/wiki/Android

[Xataka Android] ¿Qué es Android? Disponible on-line en: https://www.xatakandroid.com/sistema-operativo/que-es-android

[Developer] Aspectos fundamentales de una aplicación. Disponible on-line en: https://developer.android.com/guide/components/fundamentals?hl=es-419

[Wikipedia] Aplicación móvil. Disponible on-line en: https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_m%C3%B3vil

[Wikipedia] App Inventor. Disponible on-line en https://es.wikipedia.org/wiki/App_Inventor

[Codigo21.educacion.navarra] Primeros pasos con App Inventor. Disponible on-line en: https://codigo21.educacion.navarra.es/autoaprendizaje/primeros-pasos-con-app-inventor-2/

[Intef] Creando aplicaciones para móviles con MIT App Inventor. Disponible on-line en: https://intef.es/observatorio_tecno/creando-aplicaciones-para-moviles-android-con-mit-app-inventor-2/

Anexo I

INSTALACIÓN ARDUINO IDE PARA ESP32

I. 1 Introducción

Para poder programar la placa de control ESP32 mediante el entorno Arduino es necesario instalar previamente dicho entorno en el ordenador de trabajo, así como disponer de las librerías correspondientes para un correcto funcionamiento.

El ESP32 es un microcontrolador desarrollado por la marca Espressif [Espressif] con Wi-Fi y *Bluetooth Low Energy* incorporado. Aunque el ESP32 proporciona una gran flexibilidad y va a facilitar la expansión del IoT (*Internet of Things*) no es un microcontrolador típico que se encuentre en las placas Arduino habitualmente.

Por suerte, Espressif ha desarrollado las herramientas necesarias para integrar el ESP32 con Arduino IDE, y así poder programar en él como si de cualquier otra placa Arduino se tratase.

I. 2 Instalación de Arduino

El primer paso es tener instalado el entorno Arduino en el ordenador con el que se va a realizar el proyecto.

Como este software es totalmente gratuito y libre se puede realizar la descarga del mismo desde la página web oficial de Arduino. Se descarga la última versión para el sistema operativo necesario (Windows, Linux o Mac OS), que en este caso será Windows 10.

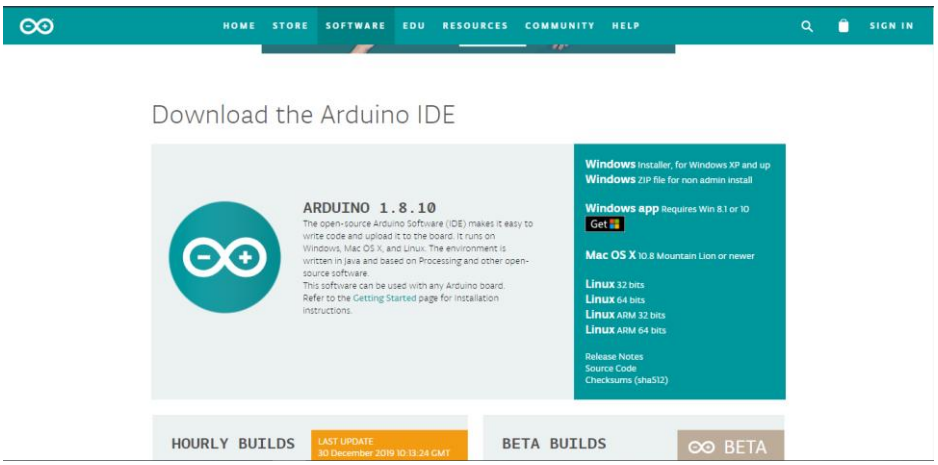

Ilustración I-1: Sitio web oficial de Arduino

Realizada la descarga, se pone en marcha el ejecutable o instalador. Se eligen los componentes que se desee que lleve incorporados y siguiendo unos sencillos pasos proporcionados por el fabricante ya se tendrá disponible Arduino IDE en el ordenador.

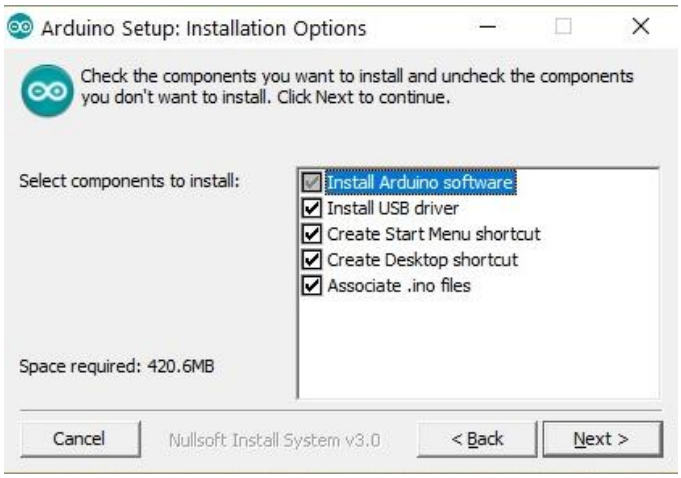

Ilustración I-2: Instalador de Arduino

I. 3 Instalación de archivos GIT

Como el entorno del ESP32 no viene integrado de serie en el Arduino IDE será necesario realizar unos pasos previos para instalar el módulo ESP32 en él y a partir de ahí configurarlo.

El primer paso ya se ha realizado que era la instalación del Arduino IDE por lo tanto procedemos al segundo paso que es la instalación de archivos de GIT. GIT es un repositorio organizado de código en desarrollo donde los programadores voluntarios van creando y depurando programas y elementos para dichos programas.

Entre ellos se puede encontrar la interfaz de Arduino para ESP32.

Descargamos e instalamos el gestor GIT desde: git-scm.com e iniciamos *Git GUI* con permisos de administrador:

- Seleccionar clonar repositorio existente.

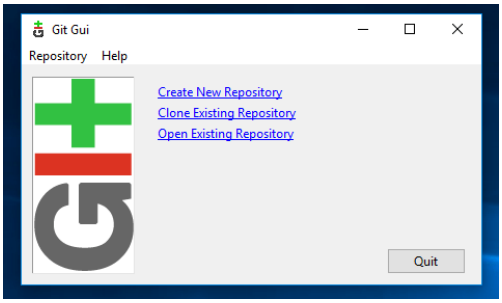

Ilustración I-3: Gestor de Git GUI

- A continuación, seleccionar fuente y destino:
  - Fuente: https://github.com/espressif/arduino-esp32.git
  - Target Directory: C:\Users\(USUARIO)\Documents\Arduino\hardware\espressif\esp32
  - Hacer click en Clone

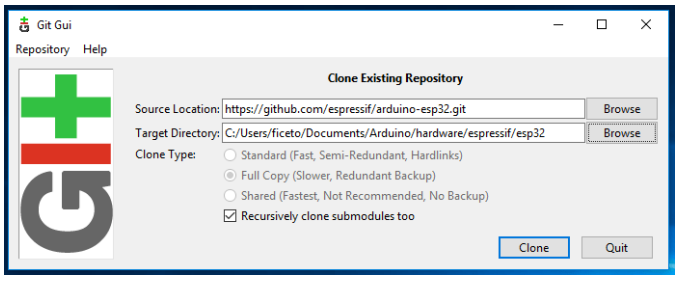

Ilustración I-4: Gestor de Git GUI

- Iniciar Git Bash apuntando /Documents/Arduino/hardware/espressif/esp32 y ejecutar git submodule update --init --recursive
- Abre *C:\Users\(USUARIO)\Documents\Arduino\hardware\espressif\tools* y ejecutar *get.exe*

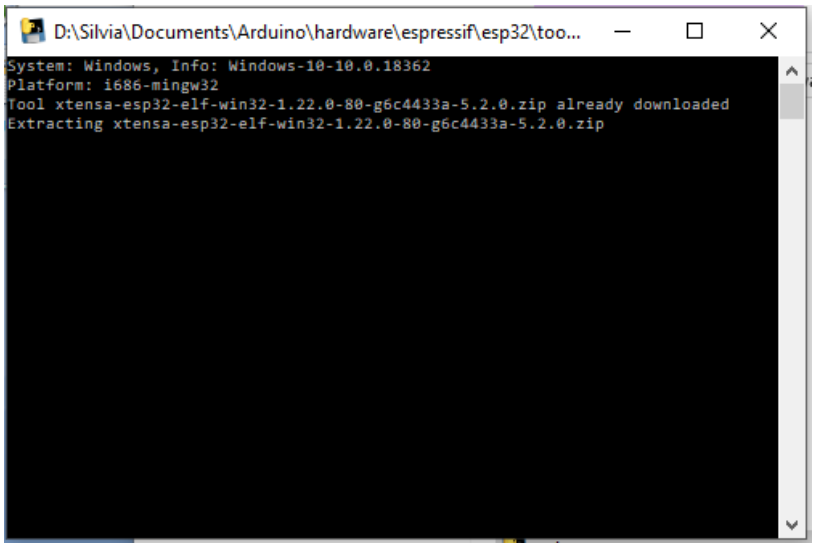

Ilustración I-5: Ejecución de get.exe

- Una vez que finalizada la ejecución de *get.exe* deberán aparecer los siguientes archivos en el directorio:

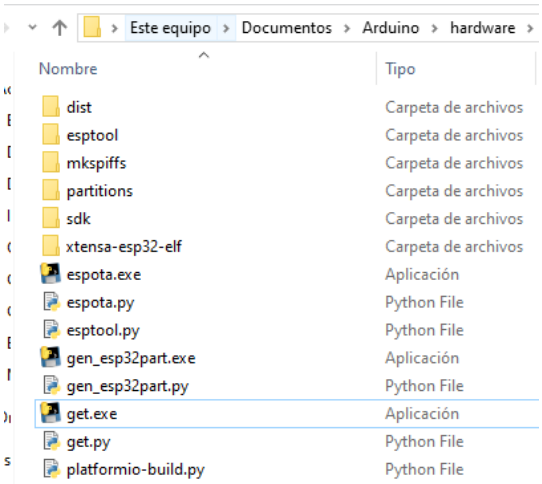

Ilustración I-6: Archivos tras la ejecución de get.exe

Una vez finalizado *get.exe* ya se puede proceder a configurar el módulo ESP32 en el entorno Arduino.

I. 4 Configuración de ESP32 para Arduino

A continuación, se procede a la configuración del módulo ESP32 en el entorno Arduino IDE.

Primero, conectar la placa ESP32 al ordenador y esperar a que se instalen los controladores (también se pueden instalar manualmente los que sean necesarios).

Una vez hecho esto, se ejecuta Arduino IDE y se selecciona el modelo de la placa en *Herramientas> placa*:

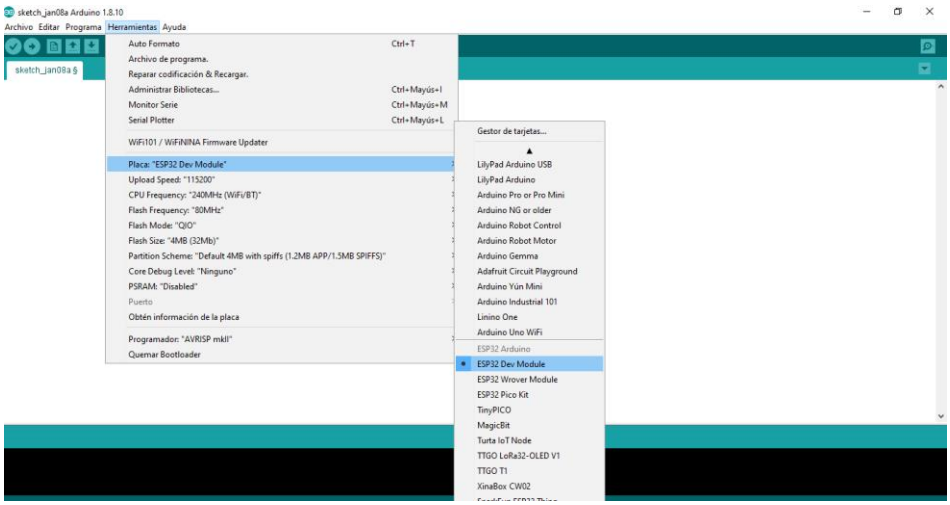

Ilustración I-7: Gestor de tarjetas Arduino

A continuación, configuramos el puerto COM de la placa. En este caso corresponde al puerto COM. Se puede saber en *Administrador de dispositivos*:

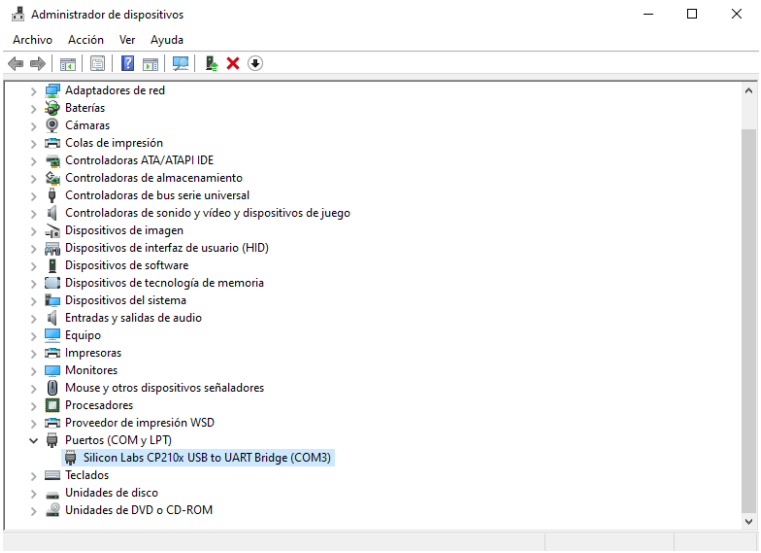

Ilustración I-8: Administrador de dispositivos

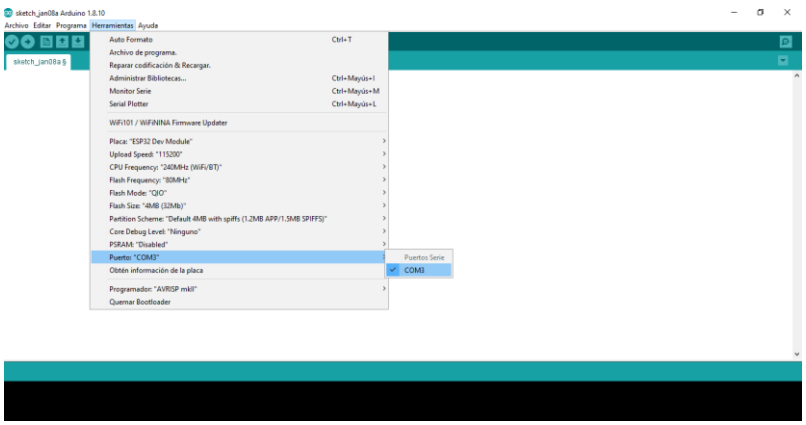

Ilustración I-9: Selección del puerto de la placa

Una vez seleccionado el puerto, tanto el entorno Arduino como la placa están preparados para el uso que se le vaya a dar.