UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Aeroespacial y Diseño Industrial

Diseño e implementación de un sistema de control electrónico de germinación de semillas en un mini invernadero

Trabajo Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

AUTOR/A: Guillem Valles, Adrian

Tutor/a: Berjano Zanón, Enrique

CURSO ACADÉMICO: 2022/2023

UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

Grado en Ingeniería Electrónica Industrial y Automática

Curso 2022/2023

TRABAJO DE FIN DE GRADO:

Diseño e implementación de un sistema de control electrónico de germinación de semillas en un mini invernadero.

DOCUMENTOS DEL PROYECTO

Documento 1. Memoria
Documento 2. Planos
Documento 3. Pliego de condiciones
Documento 4. Presupuesto

AUTOR: Adrián Guillem Vallés
TUTOR: Dr. Enrique Berjano Zanón

RESUMEN

En el mercado existen sistemas de control para invernaderos de una producción agrícola profesional o industrial. El sistema que se quiere diseñar está destinado a un invernadero de un tamaño más reducido de uso doméstico o para usuarios principiantes en la germinación y cultivo de semillas. Los objetivos de este TFG son: 1) Diseñar el sistema de adquisición de datos para las variables de temperatura, humedad, luz y nivel de agua; 2) diseñar el sistema de control del riego por goteo; 3) Diseñar el un sistema de refrigeración; y 4) diseñar una interfaz web para el control de los parámetros dependiendo de la semilla elegida. El usuario podrá controlar todos los parámetros por medio de una interfaz web. En ella se mostrarán gráficos, tablas y elementos necesarios para la correcta interpretación. Además, podrá ajustar el sistema de riego por goteo modificando las horas de riego o la cantidad de agua que se utilice.

RESUM

Al mercat hi ha sistemes de control per a hivernacles d'una producció agrícola professional o industrial. El sistema que es vol dissenyar està destinat a un hivernacle d'una mesura més reduïda d'ús domèstic o per a usuaris principiants en la germinació i el cultiu de llavors. Els objectius d'"aquest TFG són: 1) Dissenyar el sistema d'adquisició de dades per a les variables de temperatura, humitat, llum i nivell d'aigua; 2) dissenyar el sistema de control del reg per degoteig; 3) Dissenyar un sistema de refrigeració; i 4) dissenyar una interfaç web per al control dels paràmetres depenent de la llavor triada. L'usuari podrà controlar tots els paràmetres mitjançant una interfaç web. S'hi mostraran gràfics, taules i elements necessaris per a la correcta interpretació. A més, podeu ajustar el sistema de reg per degoteig modificant les hores de reg o la quantitat d'aigua que s'utilitza.

Abstract

There are control systems on the market for greenhouses for professional or industrial agricultural production. The system to be designed is intended for a smaller greenhouse for domestic use or for beginner users in seed germination and cultivation. The objectives of this TFG are: 1) Design the data acquisition system for the variables of temperature, humidity, light and water level; 2) design the drip irrigation control system; 3) Design a cooling system; and 4) design a web interface to control the parameters depending on the chosen seed. The user will be able to control all parameters through a web interface. It will show graphs, tables and elements necessary for correct interpretation. Additionally, you can adjust the drip irrigation system by modifying the irrigation hours or the amount of water used.

UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

Grado en Ingeniería Electrónica Industrial y Automática

Curso 2022/2023

TRABAJO DE FIN DE GRADO:

Diseño e implementación de un sistema de control electrónico de germinación de semillas en un mini invernadero.

DOCUMENTO 1. MEMORIA

AUTOR: Adrián Guillem Vallés
TUTOR: Dr. Enrique Berjano Zanón

Índice de contenido

Índice de figuras y tablas

1. Introducción y objetivos

Los invernaderos son sistemas fundamentales en el desarrollo y cultivo de plantas que dan lugar a los alimentos que consume la mayoría de la población. Pese a que hay gran cantidad de agricultores que se decantan por un método de cultivo más tradicional, sin un sistema de control, la producción en masa de alimentos de origen vegetal se produce en grandes invernaderos.

Los invernaderos permiten controlar todos los condicionantes que influyen en el desarrollo de la vida de una planta haciendo que esta crezca de forma saludable y en condiciones óptimas. El control de la temperatura, de la humedad relativa del ambiente, de la humedad relativa del suelo de cultivo, de la luz incidente, del riego, etc. Son algunos de los factores principales que se controlan en estas estructuras.

Este trabajo de final de grado pretende exponer el diseño del sistema de control de un invernadero de escala reducida capaz de monitorizar los niveles de temperatura, humedad y luz de la zona de cultivo, así como, controlar el riego por goteo y la refrigeración del habitáculo.

Todos los datos podrán ser observados y editados a través de una interfaz web accesible por el usuario a través de su teléfono móvil u ordenador. En la propia interfaz, se podrán editar los tiempos de riego y la cantidad de agua, comprobar el nivel de agua restante en el depósito, ajustar la temperatura correcta del invernadero en caso de temperaturas extremas, además, gráficas y tablas pueden ser utilizadas para facilitar la comprensión e interpretación de los datos.

Finalmente, se ha de recalcar los objetivos de desarrollo sostenible (ODS) que cumple este proyecto. Los ODS son unas metas globales para intentar acabar con la pobreza, proteger el planeta y asegurar una prosperidad junto con un correcto desarrollo. (Naciones Unidas, s.f.)

Este proyecto cumple el objetivo ***15. Vida de ecosistemas terrestres*** en un alto nivel y, el objetivo ***9. Industria, innovación e infraestructuras*** en menor nivel.

2. Motivación y justificación

Los agricultores no profesionales que deseen cultivar sus propias plantas desde la semilla encuentran dificultades para controlar con exactitud los diferentes factores que afectan al crecimiento y desarrollo de estas.

Existen sistemas de control para invernaderos industriales de gran tamaño que suelen ser de alto coste, pero, a nivel principiante o no profesional, no existe ningún sistema de control más sencillo, para un espacio más reducido (de unos centímetros o pocos metros) y de un

precio más asequible. Además, incorporará un sistema de monitorización que facilitará aún más la labor del agricultor principiante, pudiendo conseguir los objetivos propuesto, dando lugar a un aumento de la motivación del propio usuario.

No solo se limita a aquellos agricultores no profesionales, también está dirigido a todas aquellas entidades, escuelas, institutos o academias cuya enseñanza esté relacionada con el cultivo controlado o por invernadero ya que, con una maqueta de pequeño tamaño es más fácil mostrar los procesos que se llevan a cabo en estas estructuras.

3. Estado del arte

Desde la invención de los invernaderos, el sistema utilizado para el cultivo de plantas ha ido evolucionando a la par que el ser humano ha descubierto materiales más duraderos, resistentes, más fáciles de conseguir y métodos más efectivos.

Figura 1. Invernadero de vidrio del siglo XX

Los primeros invernaderos fueron fabricados con mica y alabastro en la época de los romanos. Posteriormente, en algunos países europeos más avanzados y asiáticos, retomaron el cultivo mediante invernaderos, esta vez fabricados a base de madera, bambú, papeles aceitados, etc. No fue hasta el siglo pasado cuando los invernaderos de vidrio tomaron popularidad. Uno de los motivos fue el gran desarrollo económico de algunos países europeos. La invención del plástico marcó un antes y un después en la fabricación de invernaderos ya que supuso una reducción del coste y un aumento en la efectividad con respecto a los invernaderos de vidrio. (Agropinos, 2021)

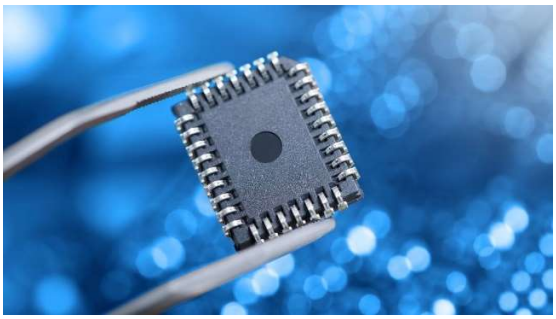


Figura 2. Microcontrolador

El siguiente paso evolutivo que permitió mejorar la calidad del cultivo de verduras y hortalizas fue la aparición del microcontrolador. En la década de 1970, tanto Intel como Texas Instruments crearon los primeros microcontroladores básicos que incorporaban una memoria, un microprocesador y un reloj. (Wikipedia, 2023) El avance en este tipo de nuevas tecnologías fue exponencial; de modo que, actualmente, se puede controlar y monitorizar cualquier aspecto que se desee dentro del invernadero. Los microcontroladores son elementos que actúan como cerebro de todo el sistema de control. Este elemento permite recibir señales a través de sensores, procesar la información y ejecutar acciones predefinidas sobre los distintos actuadores. Se forma un bucle de lazo cerrado básico el cual facilita la regulación del sistema en cualquiera de sus aspectos: control de temperatura, de humedad, de luz, del sistema de riego, etc.

Figura 3. Conexión Wi-Fi en referencia a la conexión a Internet

También adoptan una función que está cobrando gran importancia en estos últimos años: la monitorización vía Internet. Poder ajustar cualquier parámetro acerca del tiempo de riego, la activación de la calefacción o poder tener un registro de todos los datos de temperatura durante el año y poder acceder en el instante que el usuario desee solo puede conseguirse a través de un servicio web.

Actualmente, existen gran variedad de empresas que diseñan estos sistemas de una forma industrial para agricultores profesionales que cuentan con invernaderos de grandes dimensiones y cultivos extensos. Estos sistemas más complejos cuentan con gran cantidad de sensores y actuadores. La comunicación entre los distintos dispositivos a lo largo del

mismo sistema se realiza mediante una red local, accesible desde cualquier usuario que contenga las credenciales necesarias para acceder.

Estos sistemas no son asequibles para agricultores no profesionales o principiantes. Por ello es necesaria una adaptación a invernaderos de tamaño más reducido, con menor cantidad de sensores y actuadores y con un método de comunicación más sencillo que permita una rápida configuración, un montaje sencillo y un preciso control de todo el bucle.

4. Especificaciones

La zona de cultivo debe estar localizada dentro de una pequeña estructura. Albergará dos semilleros de plástico compuestos por 36 espacios para semillas cada uno. La estructura debe tener unas dimensiones que permita la manipulación de los elementos que se encuentren dentro, ya sea el equipo electrónico o los propios semilleros. La estructura estará recubierta de un plástico translúcido que permitirá el paso de la luz y, a su vez, mantendrá la temperatura estable.

El sistema electrónico irá montado en una zona externa a la zona de cultivo, pero en la misma estructura, de modo que no haya mucha distancia entre microcontroladores y sensores, pero se garantice la seguridad de los equipos de control.

El sistema requiere de unas medidas para que el usuario pueda controlar las distintas variables y ajustar los parámetros de riego, luz y temperatura dependiendo de sus necesidades.

Principalmente, se medirán 4 tipos de variables: una para la temperatura, una para el nivel de luz, una para el nivel de agua restante en el depósito y una para la humedad del suelo y poder activar el sistema de riego.
Todos los sensores serán calibrados y controlados por el microcontrolador principal. La alimentación de los sensores se realizará mediante el microcontrolador; en caso de que no fuese posible, se aplicaría una fuente externa.

El sistema contará con tres actuadores: una bomba para el sistema de riego por goteo, una fuente de luz para compensar las horas de luz que el Sol no puede proporcionar y una resistencia calefactable para épocas donde las temperaturas sean bajas.
La alimentación de los actuadores podrá ser externa en caso de que el microcontrolador no tenga la capacidad de aportar grandes cantidades de tensión o corriente; o se realizará a través del mismo microcontrolador en caso de que las tensiones y/o corrientes no sean muy elevadas. La alimentación general se realizará mediante una conexión a la red eléctrica y su correspondiente sistema de regulación adaptado a cada actuador si fuese necesario.

La interfaz estará diseñada acorde a las necesidades del usuario. Como base, contará con una tabla donde se actualizarán los valores que midan los sensores. Los actuadores también estarán registrados indicando el estado en el que se encuentren. Se deberá incluir

un modo de programación para activarlos y desactivarlos cuando el usuario lo necesite. Los valores que el usuario desee serán registrados y mostrados en el formato que se elija, pudiendo estudiar distintos factores de las semillas con las que se esté trabajando. La interfaz estará disponible en un servidor local accesible a aquellos usuarios que estén habilitados.

5. Descripción de soluciones alternativas

5.1. Estructura

El miniinvernadero va a disponer de una estructura donde se van a almacenar los semilleros y permitirá que las plantas germinen de forma adecuada. Además, dispondrá de una zona habilitada para almacenar los componentes electrónicos que se utilicen. Esta zona deberá estar protegida frente al agua y a las altas temperaturas.
La estructura será de un tamaño pequeño de modo que el usuario lo pueda tener en el balcón de su casa o en su terraza. Las dimensiones serán: 45 centímetros de largo por 35 cm de ancho por 50 centímetros de altura. La disposición de la estructura se puede observar con más detalle en el Documento 2: Planos.
Se han valorado distintos materiales para la construcción de la estructura, tales como: madera, aluminio o plástico PVC.

La madera es un elemento resistente y duradero que en cuanto a soporte se refiere sería un gran material, el inconveniente es que al mojarse pierde gran cantidad de sus propiedades y por tanto se descartó la madera como elemento de sujeción.

El aluminio es un elemento que presenta unas características que lo hacen uno de los favoritos para la construcción de los pilares de la estructura. Se caracteriza por ser un elemento ligero, contando con un peso de 2.7 g/cm3; resistente ya que, siendo un tercio más ligero que el acero, mantiene la misma resistencia; es de larga duración pudiendo llegar a durar en estructuras más de 60 años; y es económico porque comparado con elementos que se utilizan con objetivos parecidos tales como el latón o el acero inoxidable, es el más barato entre todos ellos. (Inoxidables VIctoria, 2022)

Por el contrario, el aluminio es propenso a acumular humedad debido a su impermeabilidad y su característica de ser un elemento frio favorecen la condensación de agua en la superficie del material. Además, al ser un buen conductor de la electricidad puede convertirse en un elemento peligroso si se junta con una pequeña cantidad de humedad. (Inoxidables VIctoria, 2022)

Finalmente, el aluminio se oxida, factor principal que marcará la elección del material para la construcción de la estructura.

El plástico PVC es un material resistente, fácil de tratar, económico, impermeable, de larga duración teniendo una vida útil entre 15 y 100 años en la mayor parte de sus aplicaciones en construcción, aislante térmico, eléctrico y acústico, reciclable y muchas más características que favorecen la elección de este material. (Asoven, 2018)

Como único inconveniente se podría destacar que es un elemento ligero y si se somete a grandes rachas de viento puede destrozar la estructura por completo. (Asoven, 2018)

Para proteger la zona de cultivo, va a ser recubierta con un plástico translúcido, especial para invernaderos, que permita traspasar gran cantidad de luz y manteniendo la temperatura lo más estable posible.

5.2. Sensores

La recopilación de datos es un proceso esencial para el correcto funcionamiento del sistema. Por ello, la elección correcta del sensor y el rango de medidas en el que éste trabaje será un factor determinante a tener en cuenta en el desarrollo del sistema. En este caso en particular será necesario tomar medidas acerca de la humedad en el suelo, la temperatura ambiente, la luz ambiente y el nivel de agua. Se han estudiado diferentes opciones para cada tipo de sensor pudiendo hacer así un análisis preciso que conlleve a la mejor solución posible.

5.2.1. Sensores de temperatura

La temperatura es un factor importante en el desarrollo de la vida de una planta. Hay especies que son propias de la estación de invierno y, por tanto, crecen con temperaturas bajas, incluso cercanas a 0ºC. Por otro lado, existen especies propias de las estaciones más cálidas como primavera o verano y su desarrollo se realizará con temperaturas más altas. El control de la temperatura a lo largo del periodo de crecimiento será uno de los objetivos principales del proyecto.

Existen distintos tipos de sensores de temperatura cuya diferenciación principal es el rango de temperatura en el que trabajan. Otras características a tener en cuenta serán el tamaño del sensor, la alimentación, la sensibilidad, etc.

Los principales tipos de sensores de temperatura son: elementos bimetálicos, resistencias termométricas, NTC, termopares y circuitos integrados.

5.2.1.1. Elementos bimetálicos

Los sensores de temperatura bimetálicos, tal y como su nombre indica, está formados por dos metales unidos de diferente coeficiente de dilatación que, al experimentar un cambio de temperatura se deforman. No requiere de alimentación para excitar el sensor.

Trabajan en un rango de temperaturas de entre -50ºC a 600ºC. Además de la función de sensor también pueden actuar como elementos de protección, en interruptores térmicos; y de regulación, como termostatos.

Se caracterizan por ser sensores lentos y con exactitud limitada, aunque son robustos, seguros y económicos.

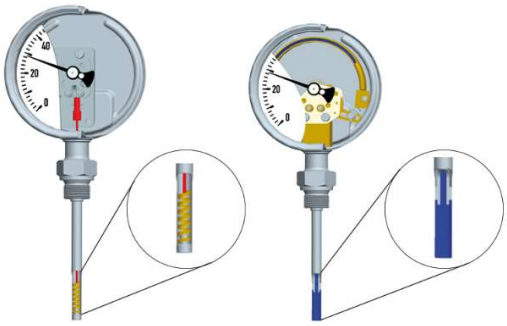


Figura 4. Manómetros analógicos formados por elemento bimetálicos

5.2.1.2. Resistencias termométricas

Las termorresistencias, también conocidas como RTD (Resistance Temperature Detector) son elementos que varían su resistencia en función de la temperatura. Pueden encontrarse modelos formados por un hilo de platino, cobre o níquel enrollado o en forma de una película fina.

Los modelos más conocidos de este tipo de sensor de temperatura son las Pt100 y las Pt1000. Este tipo se caracteriza por tener una gran linealidad y estabilidad y una alta resistividad. Los rangos de temperatura en los que trabajan varían entre los -260ºC y 1000ºC.

Figura 5. Representación de distintos modelos de termoresistencias

5.2.1.3. NTC

Los termistores, NTC (Negative Temperature Coefficient) o PTC (Positive Temperature Coefficient) son elementos que, al variar la temperatura, varía su resistencia.

Los termistores están formados por semiconductores: Manganeso (Mn), Cobalto (Co) además de los elementos nombrados previamente. Se pueden encontrar en distintas

formas tales como lentejas, barras, discos planos, etc. Los rangos de temperatura están entre los -55ºC y 450ºC.

Las RTD y los termistores comparten características en lo que a composición se refiere, pero estas últimas son más baratas, más sensibles a los cambios de temperatura y con una respuesta más rápida; mientras que las primeras son más exactas y su rango de medición es mayor.

Figura 6. Distintas formas y tamaños de sensores NTC

5.2.1.4. Termopares

Los termopares están formados por la unión de dos metales en un punto donde se mide la temperatura. Dependiendo de los metales de los que estén formados, rango de temperaturas efectivas y demás características, existen distintos modelos que son clasificados atendiendo a distintas letras del abecedario (J, K, T, E, N, …).

Los rangos de temperatura son los más extensos de todos los tipos variando entre -270ºC y 2300ºC.

Figura 7. Distintos modelos de termopares

5.2.1.5. Circuitos Integrados

Son circuitos basados en la sensibilidad frente a la temperatura de las uniones semiconductoras. Se caracterizan por ser elementos de pequeño tamaño, de bajo coste, de gran linealidad y alta sensibilidad pero que trabajan en un rango reducido de temperaturas, entre -55ºC y 160ºC. Presentan modelos tanto analógicos como digitales y salidas en tensión y en corriente.

Figura 8. Modelo TMP36

5.2.2. Humedad del suelo

La tierra es el elemento que mantiene a la planta erecta, sujeta y fija al suelo, además de aportar nutrientes esenciales para realizar la fotosíntesis. Los más importantes son: nitrógeno (N), fósforo (P), potasio (K), azufre (S), magnesio (Mg), calcio (Ca) entre otros.

El agua, por su parte, es un elemento complementario a la tierra que facilita la absorción de los nutrientes a través de las raíces. La dosis de agua proporcionada ha de ser la suficiente para que la planta realice correctamente sus funciones; una cantidad mínima o una cantidad excesiva de agua puede provocar deshidratación o sobrehidratación de la planta, respectivamente, haciendo, de este modo, que la planta muera. Además, la proporción de agua va a depender de distintos factores ambientales como la temperatura, las lluvias, la especie de planta, etc.

Los higrómetros son los sensores que se encargan de medir la cantidad de humedad que hay en el suelo. Los distintos tipos de higrómetros están explicados posteriormente.

5.2.2.1. Higrómetro capacitivo

Los higrómetros capacitivos están formados por un material especial el cual cambia su permeabilidad en función de la humedad del suelo, variando así su capacitancia. Ese valor obtenido se transforma en una señal eléctrica. Esta señal suele ser en forma de tensión así facilita la medición y el tratamiento de esta por cualquier tipo de controlador. (Restuccia, 2021)

Este tipo de sensores son robustos, duraderos y que requieren poco mantenimiento. Existen distintos formatos con diferentes alturas para poder medir a más o menos profundidad. Están destinados a sistemas a bajo nivel, sin gran complejidad y para prototipos ya que los datos obtenidos no son tan precisos y seguros como otros sensores modelos. (Ouyang, 2022)

Figura 9. Sensor de humedad capacitivo

5.2.2.2. Tensiómetros

Los tensiómetros son sensores que miden el esfuerzo que realizan las raíces para extraer la humedad del suelo, cuando la humedad sea la adecuada, la facilidad de las plantas para absorber el agua es mayor.

Está formado por tres partes: un cilindro hueco, una cápsula porosa y un vacuómetro (elemento que permite conocer la tensión dentro del cilindro). Las unidades con la que se mide son los centibares (cb). Existen tensiómetros con indicadores analógicos y con salidas digitales, los cuales proporcionan una señal eléctrica proporcional a la tensión del suelo. Las instalaciones de estos sensores suelen ir por parejas, uno situado en la zona de mayor actividad radicular y otro a mayor profundidad.

Son herramientas fáciles de utilizar e instalar y de bajo coste pese a que necesiten un mantenimiento bastante periódico y que su efectividad disminuye en suelos muy secos o con texturas muy gruesas. (SensaCultivo, 2022)

Figura 10. Sensor de humedad tensiométrico

5.2.2.3. TDR (Time Domain Reflectometry)

Los sensores TDR usan un oscilador para generar señales de alta frecuencia. Estas señales son transmitidas a un elemento metálico y este, al suelo. La amplitud de la señal de retorno al sensor indicará el nivel de humedad. Este valor se le conoce como constante

dieléctrica y actúa proporcionalmente al nivel de humedad en el suelo, es decir, a medida que aumenta la humedad, la constante dieléctrica aumenta y viceversa.

Son sensores robustos, sin un gran mantenimiento, muestran resultados independientemente del tipo de suelo, densidad o temperatura además de ser capaces de medir a temperaturas extremadamente bajas. Por el contrario, son elementos caros y complejos en su diseño. (SensaCultivo, 2022)

Figura 11. Sensor de humedad TDR

5.2.2.4. FDR (Frequency Domain Reflectometry)

Este tipo de sensores aparecieron por la década de los 80, simplificando el funcionamiento de los TDR. Se basan en pulsos electromagnéticos propagados en un medio. La constante dieléctrica se mide a partir de la frecuencia media obtenida de las ondas electromagnéticas propagadas.

Los sensores FDR son más rápidos, precisos, simples, seguros, con un rango más amplio de mediciones y con una calibración más simple ya que el tipo de suelo no afecta al sensor. (SensaCultivo, 2022)

Figura 12. Sensor de humedad FDR

5.2.3. Luxómetro

La luz es uno de los factores más importantes en la vida de la planta, en concreto, en el proceso de creación de nutrientes con la que la planta se alimenta, la fotosíntesis. La planta utiliza la clorofila que está en sus hojas para captar la luz del sol.

Según la época del año las horas de luz de sol varían. Las plantas más comunes de verano reciben más cantidad de luz, pero, debido a la época donde se desarrollan, necesitan más cantidades de luz. Al revés ocurre con las plantas típicas de invierno, reciben menos cantidad de luz, pero, a su vez, necesitan menos horas de luz. Además, hay factores meteorológicos que alteran las horas de luz como los días nublados o la noche. Por este motivo es importante registrar la cantidad, el nivel y el tiempo que la luz incide sobre las plantas.

Los sensores lumínicos o luxómetros son los indicados para realizar este tipo de mediciones. Existen cuatro tipos: los fotodiodos, los fotorresistores (LDR), los fototransistores y las células fotoeléctricas. (Mecatrónica LATAM, 2021)

5.2.3.1. Fotorresistores (LDR)

Los fotorresistores o LDR (Light Dependent Resistor) son dispositivos que se utilizan para medir la variación del nivel de luz mediante la variación de la resistencia del sensor.

El fotorresistor está formado por un semiconductor, normalmente sulfuro de cadmio (CdS). Al incidir la luz sobre él algunos de los fotones son absorbidos, provocando que electrones pasen a la banda de conducción y, por tanto, disminuyen la resistencia del componente. Esta variación en la resistencia del componente es el primer indicador de cambio en la luz incidente sobre el sensor.

Son elementos baratos y fácil de adquirir, pero debido a si baja precisión y la alta dependencia con la temperatura, les hacen no ser los sensores adecuados para actuar como luxómetros. (Feria de tecnologia, s.f.)

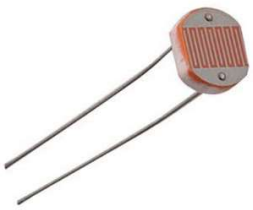


Figura 13. Fotorresistencia LDR

5.2.3.2. Fotodiodos

El fotodiodo es un diodo semiconductor de tipo PN sensible a la luz infrarroja y visible. El funcionamiento del fotodiodo es bastante simple ya que dependiendo de la luz que reciba el diodo, el flujo de corriente que pasa por dentro de él variará haciendo que la tensión de

salida también varíe. Cuanto mayor sea la intensidad de la luz, mayor será el flujo de corriente.

Estos sensores adoptan un comportamiento más lineal que las LDR y la respuesta frente a los cambios es más rápida. Al igual que con las LDR son elementos baratos y fáciles de conseguir. (Menna, s.f.)

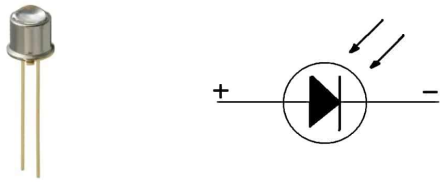


Figura 14. Fotodiodo y su símbolo eléctrico

5.2.3.3. Fototransistores

Los fototransistores son diseñados a partir de un transistor NPN donde, mayoritariamente, la base queda expuesta a la luz a través de una carcasa. Como gran parte de los transistores, puede utilizarse como amplificación de corriente ya que la que es proporcionada es entre 50 y 100 veces mayor que la que puede suministrar un fotodiodo.

Existen varias configuraciones de fototransistores. Aquellos que solo cuentan con dos pines, están formados por el colector y el emisor, mientras que la base se sustituye por el valor que es capaz de medir a través de la carcasa. Aquellos con tres terminales, sí cuentan con el pin de la base, que puede ser utilizado o no dependiendo del uso que se le quiera dar.

Además, están diseñados para actuar frente a un gran rango de tipos de luz como, por ejemplo, la luz visible, la ultravioleta, la infrarroja, etc.

Figura 15. Fototransistor

5.2.4. Sensor de nivel

El agua es uno de los nutrientes más importantes de la alimentación de las plantas y más aún en su época de germinación. El invernadero que se va a diseñar va a contar con un sistema de riego por goteo el cual estará formado por un tanque de agua, una bomba de succión, un conjunto de mangueras de riego y un sensor de nivel.

Este apartado se centra en el último elemento comentado ya que será necesario controlar el nivel de agua que contenga el tanque de modo que el regado de las semillas no se vea interrumpido bajo ningún concepto. Además de la medición del nivel, ya sea de líquidos o de sólidos, ciertos sensores reaccionan cuando el elemento llega al límite del nivel. Esta interrupción se realiza mediante un contacto de conmutación.

Por otro lado, los sensores cuya función es la medición únicamente, pueden generar señales tanto analógicas como digitales, de tensión o de corriente, en diferentes magnitudes absolutas y relativas o en distintas unidades.

El objetivo del proyecto es reducir al máximo los costes y la complejidad de los sistemas por lo que comentar aquellos sensores más complejos no aporta mucho. En los siguientes apartados se comentarán aquellos sensores de nivel con el comportamiento más simple y accesible para los usuarios a los que se dirige el sistema. (Wikipedia, 2023)

5.2.4.1. Sensores capacitivos

Los sensores capacitivos utilizan un campo eléctrico para medir distancias. Su principio de funcionamiento se basa en la variación de la capacidad eléctrica de un condensador. El campo eléctrico se crea gracias a un oscilador que contiene el propio sensor entre dos placas o electrodos. Al acercar un elemento metálico o no metálico, la capacitancia del condensador varía. El otro factor que puede afectar a la variación de la capacitancia es la constante dieléctrica del elemento que se está midiendo.

Resumiendo, la capacidad del condensador depende de la distancia, el área de las placas del sensor y de la constante dieléctrica; y viene dada por la siguiente fórmula:

$$C = \varepsilon * \frac{d}{A}$$

Aplicando la fórmula, la capacitancia aumentará según aumente la distancia al elemento que se quiera medir, según aumente la constante dieléctrica o según disminuya el área de medición del sensor.
Además, este tipo de sensores llevan incorporados un potenciómetro para ajustar la sensibilidad de modo que, si se varía el elemento a medir y, por ende, la constante dieléctrica, no sea necesario cambiar de posición el sensor en la instalación ya realizada.

La principal ventaja de este tipo de sensores es la amplia variedad de líquidos y sólidos en los que se pueden utilizar, incluyendo elementos metálicos y no metálicos.
El principal inconveniente es la suciedad ya que pueden llegar a afectar a la capacidad del condensador, aunque son fáciles de limpiar.

Figura 16. Sensor capacitivo

5.2.4.2. Sensores ultrasónicos

Los sensores de nivel de ultrasonidos utilizan ondas ultrasónicas para la detección de nivel de líquidos y sólidos sin contacto. El principio de medición se basa en el tiempo transcurrido entre que el emisor manda la onda y el receptor la recibe.

Aunque son sensores que se ven afectados por factores externos como la humedad, la temperatura, la presión, vapores, espumas, etc. Los resultados que se obtienen son bastante fiables. De cualquier forma, existe la posibilidad de añadir un filtrado de señal analógico o digital para el tratamiento de esta, facilitando así el proceso al microcontrolador que tenga que interpretar la señal.

Otro factor que interviene en la correcta interpretación de la señal y que no depende de un proceso electrónico es el montaje. El sensor ha de estar situado en un lugar libre de obstáculos para que la señal sea lo más limpia posible.

Finalmente, son sensores baratos, fáciles de conseguir y con resultados rápidos y fiables. El sensor más conocido de ultrasonidos es el HC-SR04.

Figura 17. Sensor ultrasonidos HC-SR04

5.3. Control tira de LEDs

La función principal de la tira de LEDs es sustituir la luz del sol en aquellas épocas del año en las que las plantas no reciben la suficiente energía solar para llevar a cabo la fotosíntesis.

Dependiendo de la hora del día en la que se active la luz, los leds funcionarán a distinta intensidad. Para ello, se ha de diseñar un control que permita, a partir de una o varias señales, regular la intensidad de la luz.

El planteamiento inicial consiste en implementar un componente o circuito que sea capaz de gestionar la intensidad que necesita la tira de leds partiendo de una salida digital o analógica de la tarjeta Arduino Mega.

Como ideal inicial, se había planteado un circuito formado por tres transistores BJT con distinta beta de modo que, al aplicar un valor distinto de corriente por la base, la corriente por el emisor sea distinta. El parámetro beta de cualquier transistor bipolar indica la eficiencia del transistor relacionando la corriente del colector con la de la base. Cuanto mayor sea el valor de beta, mayor será la eficiencia del transistor, es decir, con una corriente más pequeña en la base, será capaz de entregar una corriente mayor en el colector. Este parámetro viene dado por la siguiente fórmula:

$$\beta_F = \frac{I_C}{I_B}$$

Siendo I_C la corriente del colector y I_B la corriente de la base.
La alimentación de los transistores sería la misma para los tres, es decir, estarían conectados en paralelo, y, por el emisor, los tres conectados a la tira de leds. Funcionarían a modo de interruptor.

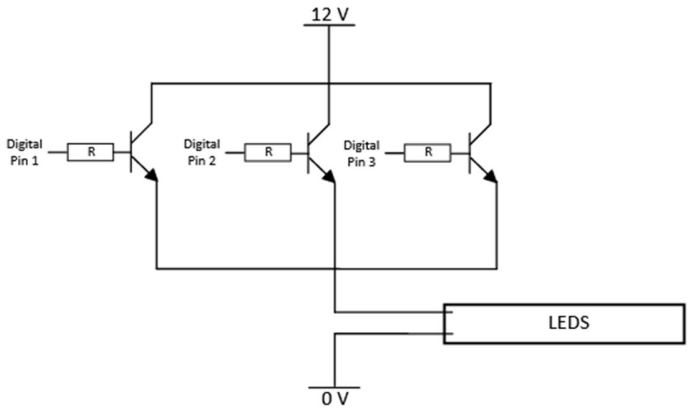


Figura 18. Triple transistor en paralelo

Este método tiene varios inconvenientes que dificultan el correcto funcionamiento. El primer inconveniente es la dificultad para encontrar los transistores con una beta tan variada. En segundo lugar, el transistor estaría funcionando en su zona activa, no en modo conmutación y, por lo tanto, el consumo y la disipación de calor serían bastante elevados. En tercer y

último lugar, los tres transistores en paralelo podrían provocar interferencias electromagnéticas afectando en el proceso de conmutación de cada uno de ellos.

Visto queda que el método anterior no es sencillo de montar ni efectivo en su funcionalidad.

El segundo método propuesto reduce el número de transistores a uno solo y conecta las tres entradas digitales procedentes del microcontrolador a la base del transistor. La idea de funcionamiento es similar a la del método anterior.

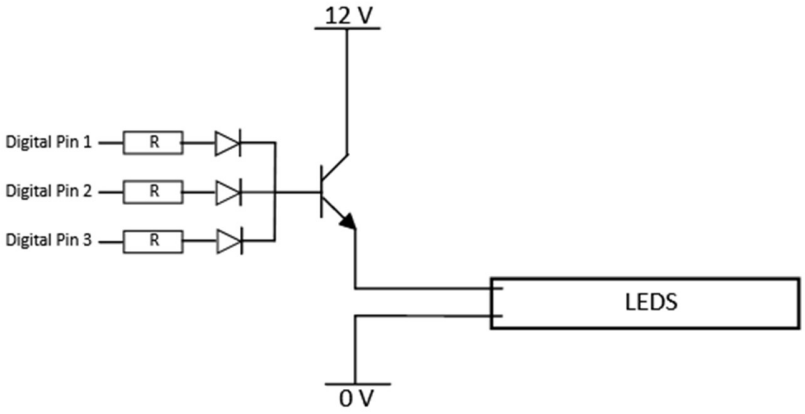


Figura 19. Transistor controlado por tres entradas digitales

Solamente una de las tres salidas digitales estaría activa y, mediante la resistencia y el diodo, se configurarían tres valores de corriente equivalentes a 25%, 50% y 75% de intensidad.

El principal problema que se encuentra en este método es que la señal digital en estado bajo no significa que no está aportando corriente, sino que la está absorbiendo, de modo que, aunque se indique que la señales no están activas, sí lo están por lo que no se puede controlar la tira de leds con un único transistor.

La tercera opción que se plantea es el control mediante una señal PWM y un transistor MOSFET. La modulación por ancho de pulso a altas frecuencias permite una elevada conmutación del transistor con lo que, variando el ciclo de trabajo, se podrá controlar la cantidad de corriente que se le transmite al MOSFET y a su vez, a la tira de leds.
En la gráfica se observa el comportamiento que se ha explicado anteriormente.

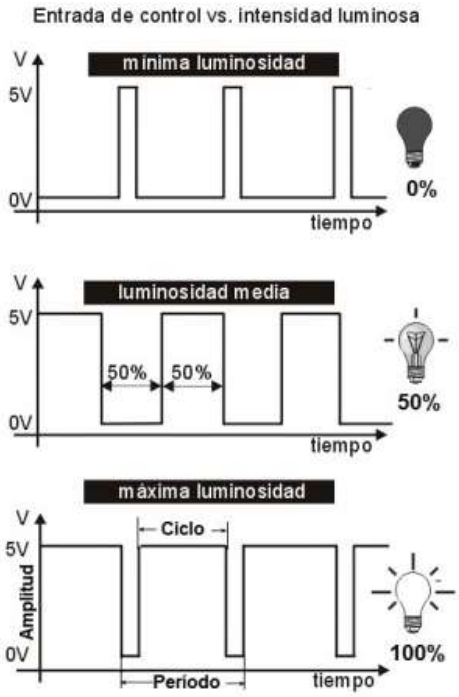


Figura 20. Control de la intensidad de la luz mediante modulación por ancho de pulso (PWM)

El diagrama es igual de sencillo que las opciones anteriores y, a priori, no presenta gran cantidad de inconvenientes.

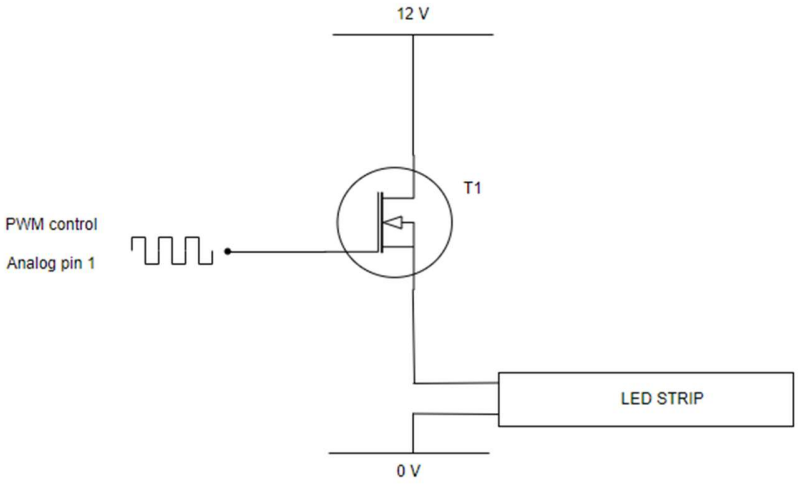


Figura 21. Tira de LEDs controlada por PWM mediante un MOSFET

6. Descripción de la solución adoptada

6.1. Estructura

El material que se ha decidido para construir la estructura donde se cultivarán las semillas es el PVC por las características comentadas en el apartado 5.1. La estructura estará recubierta por un plástico específico para invernaderos. La decisión de dicho material se ha tomado en base a unos conocimientos básicos; para una elección más adecuada, se deberá realizar un estudio más adecuado y específico que no se va a contemplar en este trabajo. Los planos de la estructura se encuentran con más detalle en el Anexo I.

Los componentes del plástico PVC van a ser de 20 mm de diámetro. Se va a contar con 6 m de tubo recto, 4 codos de 45º, 2 codos de 90º y 10 tapones hembra. Antes de comenzar el montaje, se van a cortar los 6 m de tubo recto en 4 trozos de 35 cm, 4 trozos de 25 cm, 2 trozos de 34 cm, 2 trozos 48 cm y 3 trozos de 45 cm.

En la siguiente figura se muestran todos los elementos que van a formar parte ya cortados y ordenados.

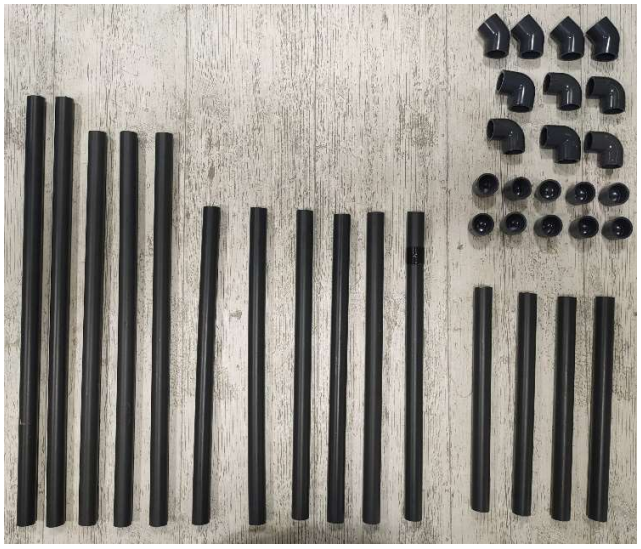


Figura 22. Material PVC cortado según las medidas especificadas.

El primer paso para comenzar con el montaje es unir las esquinas, es decir, los codos de 45º y 90º junto con los tapones hembra para conseguir una forma de “T”. La unión se realizará mediante silicona transparente especial para este tipo de material. En la siguiente figura se observa como quedan montadas todas las esquinas.

Figura 23. Codos de 45º y 90º unidos a un tapón hembra formando una “T”

Una vez estén listas las esquinas, se pueden unir los trozos rectos. Se empezará uniendo el triángulo que formará el tamaño del techo. Se necesitarán dos triángulos, uno por cada lado.

Figura 24. Triángulos que forman el techo de la estructura

El techo de la estructura quedará completo uniendo los tres trozos rectos de 45 cm.

Figura 25. Conexión del techo de la estructura

Posteriormente, en el lado de la conexión en “T” que esté libre, se conectará el trozo recto de 35 cm en cada una de las cuatro esquinas, dándole así, altura a la estructura.

Figura 26. Esqueleto del invernadero

Finalmente, para darle más robustez a la estructura se van a conectar dos trozos rectos de 34 cm y dos de 48 cm en la parte baja formando un rectángulo. De este modo se evitan que los pilares pierdan estabilidad y se pueda derrumbar.

Figura 27. Base del invernadero para aportar robustez

Una vez el esqueleto de la cabaña esté montado y se ha asegurado que las uniones son fuertes, se puede forrar la estructura con el plástico especial para invernaderos. Se medirán y cortarán dos rectángulos para las paredes más grandes, dos rectángulos para las paredes

más pequeñas, dos rectángulos para la parte del techo y dos triángulos para el techo. Las dimensiones correspondientes de cada una de las partes están especificadas en el Documento 2: Planos.

6.2. Diagrama de bloques

La siguiente imagen muestra un diagrama de bloques donde se puede observar de un simple vistazo el funcionamiento general del sistema. Cada uno de los sensores están dirigidos hacia el microcontrolador ya que únicamente transmiten información, no hay ningún tipo de comunicación desde el microcontrolador a los sensores. Lo mismo ocurre con los actuadores, pero en sentido contrario que al de los sensores; desde el microcontrolador a los actuadores. Por último, las comunicaciones entre ambos microcontroladores, el servidor y el usuario son de ambos sentidos.

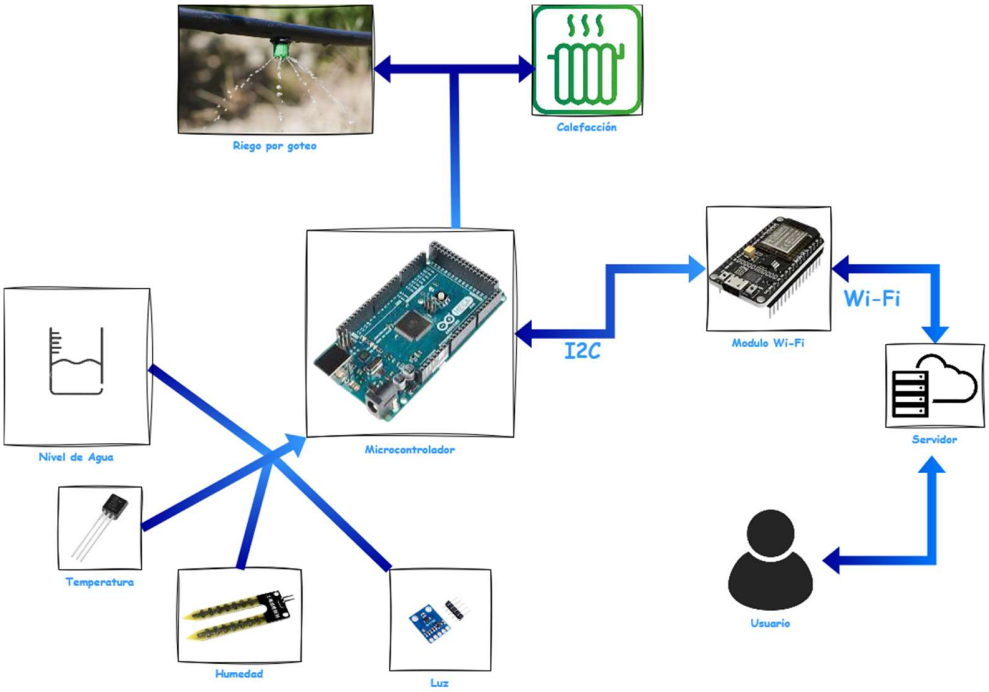


Figura 28. Diagrama de bloques

6.3. Microcontroladores

6.3.1. Arduino Mega

Arduino es la marca de elementos electrónicos por excelencia si hablamos de iniciación en el mundo de la electrónica. Está basada en un hardware y software libre, fácil de utilizar. Dispone de gran cantidad de modelos que se ajustan a las necesidades del usuario. Además, tiene una gran variedad de módulos externos a las placas con los microcontroladores que sirven de accesorios para realizar una infinidad de proyectos. (Fernández, 2022)

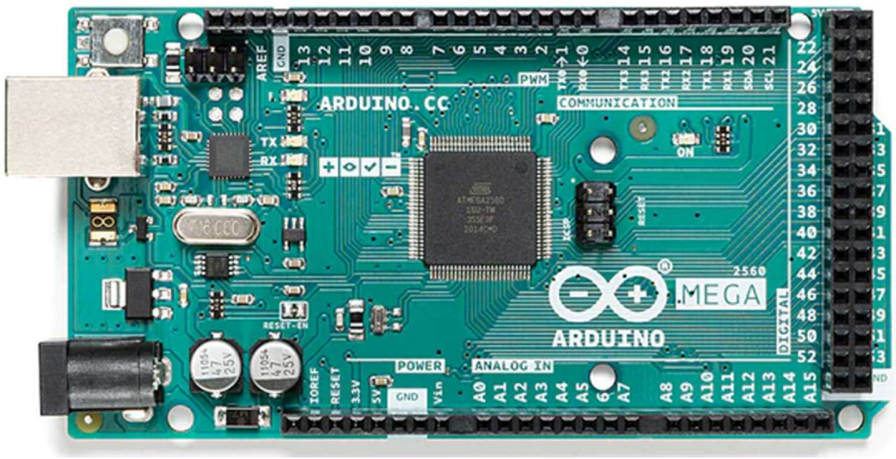


Figura 29. Arduino Mega 2560 Rev3

Se han planteado dos modelos para llevar a cabo el proyecto. Las características principales en la que se ha basado para elegir el modelo son la cantidad de pines analógicos y digitales que ofrecía la placa y algunos aspectos acerca de las capacidades de la memoria.

	Arduino Uno	Arduino Mega
Procesador	ATmega328p	ATmega2560
Velocidad	16 MHz	16 MHz
Memoria flash (kB)	32	256
SRAM (kB)	2	8
Pines digital E/S	14	54
Pines digitales con PWM	6	15
Pines analógicos	6	16

Tabla 1. Comparación entre Arduino UNO y Arduino Mega

Se puede observar que la placa Arduino Mega presenta mayor cantidad de pines digitales y analógicos, por lo que la hace ideal para el propósito del proyecto. Todos los sensores y actuadores, así como la conexión con módulos externos se podrán realizar correctamente. (Arrow, 2017)

6.3.2. ESP8266

El módulo ESP8266 es componente que es capaz de conectarse a una red Wifi y por tanto realizar gran cantidad de funciones que requieran una conexión a internet. Existen otros modelos como el MKR1000 de Arduino o ATSAMW25 de Atmel que realizan funciones similares pero la gran diferencia entre ellos es el precio. El módulo ESP8266 se puede encontrar por unos 3€, mientras que los otros dos modelos superan los 30€. (del Valle Hernández, 2018)

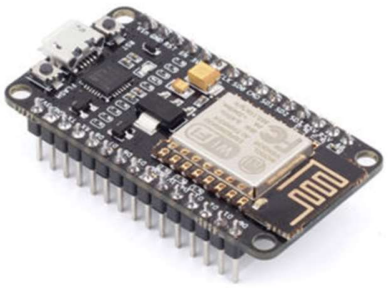


Figura 30. Módulo ESP8266

Dentro del mundo del ESP8266 existen versiones más básicas y antiguas que han ido evolucionando a lo largo de estos últimos años. El más novedoso es el ESP-12 cuya ventaja principal es su bajo consumo. Presenta también un tamaño bastante reducido lo cual permite la fácil maniobrabilidad. Dispone de pines digitales de entrada y de salida, de pines de alimentación externa tanto de entrada como de salida, llegando a suministrar hasta 3.3V; tiene un par de pines para comunicación serial y los pines necesarios para la comunicación I2C. La diferencia con otros modelos es la cantidad de pines y el rendimiento que es capaz de ofrecer el procesador, así como los tamaños en los que se puede encontrar el módulo. (Naylamp Mechatronics, 2016)

Otro punto a favor de esta familia de microcontroladores es su compatibilidad con el entorno de programación de Arduino, Arduino IDE. Basta con instalar la librería necesaria y configurar correctamente el puerto serial del ordenador para la comunicación con la placa.

6.4. Sensores

6.4.1. Temperatura

Atendiendo a las características previamente descritas y a las necesidades del proyecto se ha elegido un circuito integrado como sensor de temperatura ya que, al no tener mucho espacio dentro de la estructura, es la solución más eficaz y sencilla para el proyecto.

El sensor elegido es el *TMP36*. El rango de medida de temperaturas varía entre -40ºC y 125ºC y se alimenta con tensiones entre 2.7V y 5.5V; un voltaje que puede ser proporcionado por el propio microcontrolador. Se alimenta con una corriente de unos 65 mA con lo que tampoco supone un problema en lo que a consumo se refiere. Tiene una resolución de 10mV/ºC.

Como la lectura se va a realizar mediante el pin analógico será necesario una operación de conversión para mostrar los datos en grados y no en voltios. La fórmula es la siguiente:

$$Temperatura(℃) = \frac{5.0V}{1023.0} \cdot \frac{1°C}{10mV}$$

El primer valor corresponde con el voltaje máximo que el pin analógico es capaz de leer (5 V), el valor 1023 es la resolución máxima que el pin puede procesar y la última operación es el factor de conversión del sensor. Se han realizado pruebas previas a su montaje para asegurar su correcto funcionamiento y ajustar la configuración de este.

En la figura 31 se observa el sensor conectado al microcontrolador mediante un cableado bastante simple. Solo se disponen de 3 cables: el rojo es la alimentación (5 V), el negro es la conexión a negativo (0V) y el amarillo es la conexión de lectura al pin analógico (A0) del Arduino Mega.

Esta conexión se realiza a modo de prueba y calibración. Una vez montado el sensor en el invernadero, la única variación será la longitud del cable.

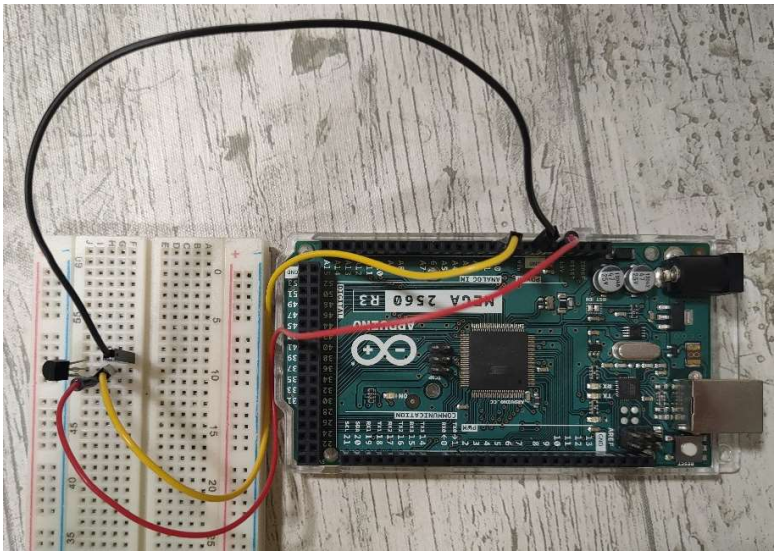


Figura 31. Conexiones entre el microcontrolador y el sensor TMP36

En la figura 32 se observan los valores que va midiendo el sensor tanto el valor digital que obtiene el microcontrolador directamente desde su pin de lectura como su correspondiente valor en la unidad de temperatura que corresponda, en este caso, grados centígrados (ºC).

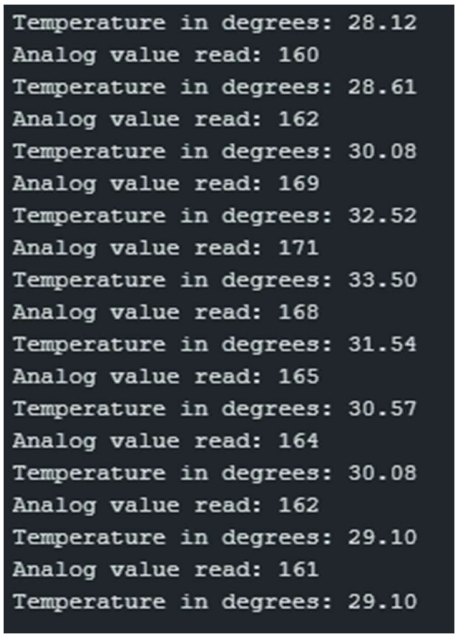


Figura 32. Lectura del sensor de temperatura y su conversión a grados centígrados

Se puede observar que los primeros valores muestran la temperatura de una sala. Para comprobar la variación de temperatura, se ha aplicado calor al sensor simplemente acercando los dedos y, por tanto, se ve un aumento en la temperatura que se lee. Progresivamente, el sensor vuelve a estabilizarse a la temperatura ambiente.

Una vez asegurado el correcto funcionamiento se puede implementar tantos sensores de temperatura como sean necesarios en el proyecto.

6.4.2. Sensor de nivel

Tras analizar todas las opciones, la opción elegida para medir la cantidad de agua que contiene el tanque es un sensor de ultrasonidos. El modelo es el *HC-SR04*, típico sensor de los packs de Arduino, formado por un controlador propio, un generador de ondas y un receptor. Se alimenta con 5V, el consumo es de 75 mW y se comunica con el microcontrolador a través de dos pines: el pin “Echo” y el “Trigger”, uno de entrada y el otro de salida, respectivamente.

El pin echo envía la señal al controlador del sensor para que este envíe la señal ultrasónica. Una vez el sensor reconozca la onda de vuelta, el pin trigger envía los resultados al microcontrolador principal. Tomando la velocidad del sonido (3400 m/s) se puede calcular la distancia de la siguiente forma:

$$Distancia\ (cm) = \frac{tiempo \cdot 0.000001 \cdot Vel\ Son}{2}$$

Se utilizará una garrafa de 6 litros de capacidad como tanque de agua. El sensor irá situado en el propio tapón de la garrafa apuntando hacia abajo, es decir, hacia el agua. El sensor de ultrasonidos es capaz de medir la distancia a la que se encuentra el agua de modo que

conociendo la altura máxima y mínima que el tanque puede almacenar, se podrá tener controlado el nivel de agua.

La garrafa de 6 litros tiene una altura de 30 centímetros. Cuando el sensor mida la distancia mínima al agua se considerará como que el tanque está lleno mientras que si el sensor recopila un valor cercano a la altura de la garrafa indicará que estará vacía. Si la cantidad de agua es mínima, se inhabilitará la bomba de riego y se notificará al usuario para que rellene el tanque y pueda realizarse el riego.

Para comprobar el correcto funcionamiento del sensor, en la figura 33 se ha representa el cableado del microcontrolador con el sensor de ultrasonidos. El pin de alimentación (5V) corresponde con el cable de color rojo, el punto negativo es el de color negro, el pin verde es la conexión con el pin Trigger y el amarillo con el pin Echo, ambos pertenecientes al sensor. Por la parte del microcontrolador, se han conectado a dos pines que funcionan como entradas digitales.

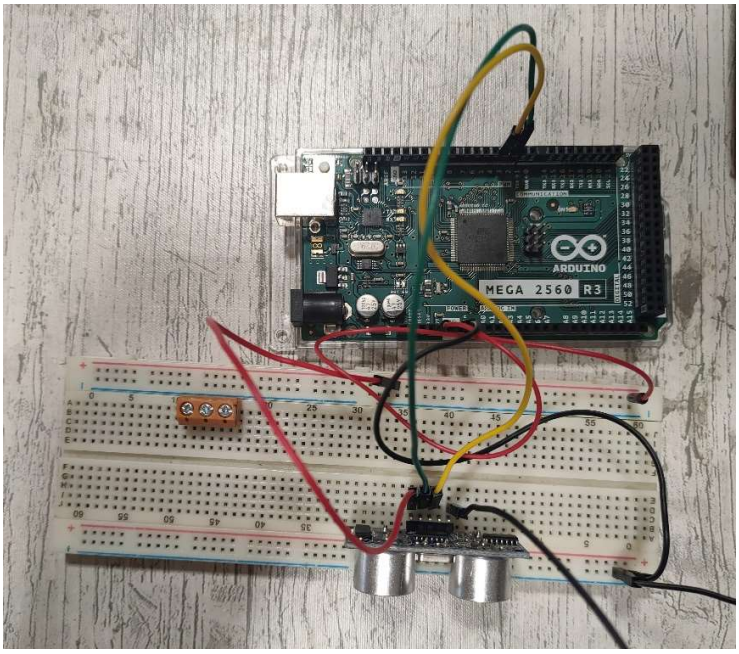


Figura 33. Cableado entre el sensor de ultrasonidos y el microcontrolador

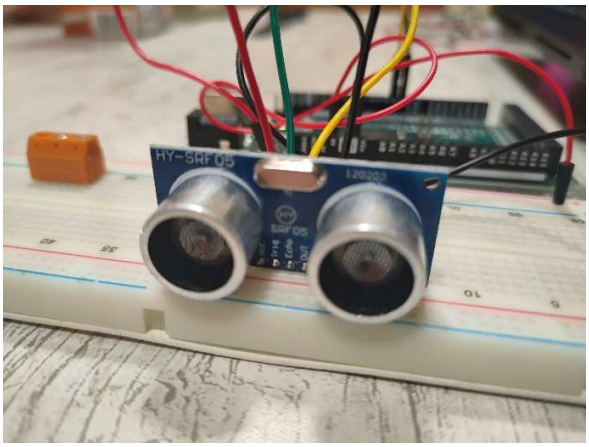


Figura 34. Sensor ultrasonidos HC-SR04

Además del montaje se ha comprobado que las lecturas son correctas utilizando una regla métrica y acercando y alejando un objeto para variar la distancia. En la figura 35 se observan los valores recopilados.

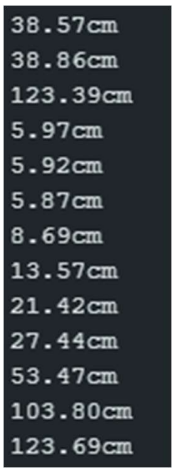


Figura 35. Valores obtenidos del sensor de ultrasonidos

6.4.3. Humedad

El sensor de humedad es el modelo *Hygrometer V1.2*. Es un modelo sencillo formado por la parte capacitiva que irá incrustada dentro de la tierra, un procesador de señal y tres pines, dos de alimentación (uno para los 5V y otro para el negativo) y uno para la salida analógica de las lecturas. La señal analógica de salida tiene como rango entre 0 voltios y 5 voltios. El valor más alto (5V) corresponde con la humedad más baja posible, mientras que el rango más bajo (0V) corresponde con la unidad máxima.

Este sensor servirá para indicar al usuario cuando tiene que encender el riego por goteo ya que se necesitará humedecer el suelo. Por otro lado, si el usuario lo requiere, se puede añadir una configuración que permita automatizar el riego, no solo por el tiempo predeterminado, sino por la cantidad de humedad que exista en el suelo.

Se han realizado pruebas previas al montaje del sensor en la estructura para comprobar su funcionamiento y su medición, ya que la hoja de datos proporcionaba poca información al respecto. En la figura 36 se muestra la conexión del sensor junto al microcontrolador principal, la cual es bastante sencilla. El cable rojo corresponde con la alimentación (5V), el cable negro es la conexión con el negativo (0V) y el cable amarillo se conecta al pin analógico (A1) del Arduino Mega.

Una vez montado todo el sistema, se van a incluir hasta 4 sensores. La principal diferencia con el cableado de la figura 36 es la longitud de los cables.

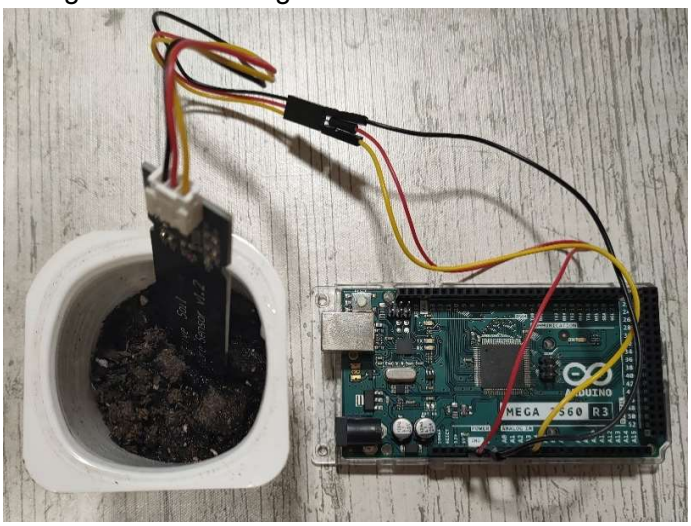


Figura 36. Conexiones entre el microcontrolador y el sensor de humedad

En la figura 37 se muestran los valores obtenidos por el sensor. Como se ha comentado anteriormente, los primeros valores que se miden corresponden a la tierra seca, con poca humedad mientras que según avanzan los valores, el rango de humedad baja según se va regando y añadiendo humedad a la tierra.

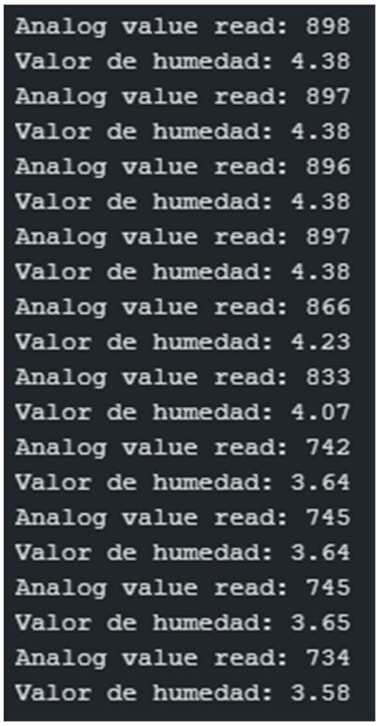


Figura 37. Lectura del sensor de humedad

6.4.4. Luxómetro

El sensor de luz es el modelo *BH1750*. Es una pequeña tarjeta que incorpora su propio controlador y procesador de señal. El método para captar la luz es mediante fotodiodos. La comunicación con el microcontrolador principal es digital a través de los pines SDA y SCL de ambas tarjetas, tanto del sensor como del microcontrolador. Se alimenta con 5V ya que en el propio módulo tiene un regulador de tensión que reduce ese voltaje de entrada a 3,3V necesarios para alimentar el procesador de señales del sensor. Consume una corriente de 0.2 mA, la cual no supone un gran problema para el sistema.

La medición de la luz se realiza a través de una función implementada en el propio código del sensor que puede ser utilizada mediante la inclusión de la librería en el código principal. El rango de valores que el sensor puede medir va desde los 0 lux, es decir, cuando el sensor no está percibiendo ninguna fuente de luz; hasta los 62000 lux aproximadamente, valor indicando la máxima luz posible que está detectando el sensor.

En la figura 38 se muestra la conexión del módulo del sensor con la tarjeta del microcontrolador. El cable rojo es la alimentación del sensor (5V), el cable negro y el verde se conectan al punto negativo (0V), el cable amarillo se conecta con el pin SCL del Arduino Mega, el cable azul con el pin SDA.

El cable verde requiere de la conexión a un nivel bajo para la correcta configuración del sensor. Sería posible conectarlo a un nivel alto con la peculiaridad que en el código de

configuración del módulo se tendría que indicar. Las conexiones de los pines SCL y SDA permiten la comunicación digital entre el microcontrolador y el módulo del sensor.

En la figura 39 se muestra con más detalle los colores de los cables con los pines del módulo *BH1750*.

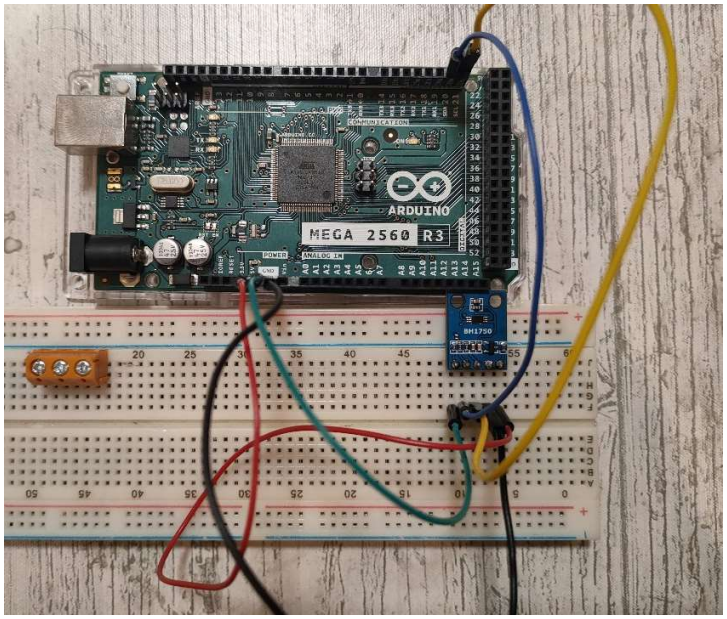


Figura 38. Conexiones entre el microcontrolador y el luxómetro BH1750

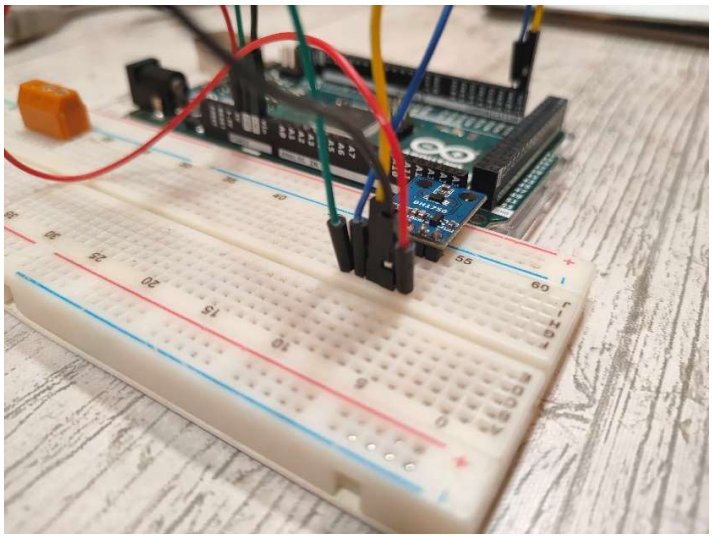


Figura 39. Cableado en el sensor BH1750

En la figura 40 se muestran los valores tomados por el sensor. Los primeros valores corresponden con una luz LED blanca situada a 2 metros de altura, posteriormente se apaga dicha luz para comprobar que el valor mínimo corresponde con la oscuridad y finalmente, se enciende una luz halógena de un color más cálido situado a la misma altura

que la luz blanca, pero con menor intensidad y se observa que mide correctamente la variación de luminancia.

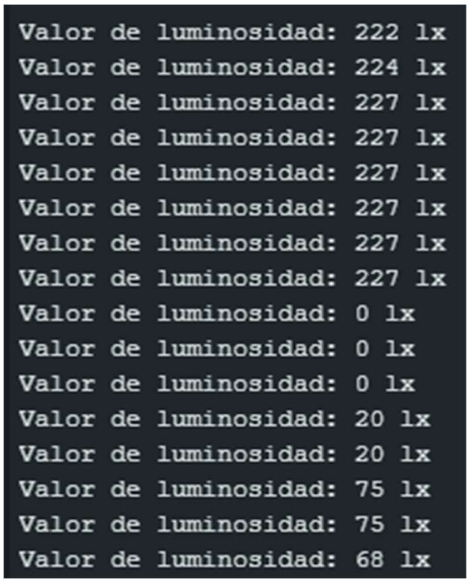


Figura 40. Lectura del sensor de luminosidad

6.5. Actuadores

6.5.1. Bomba de agua

Para poder llevar a cabo el riego por goteo será necesario obtener el agua del tanque. Se va a utilizar una bomba de agua sumergible con un caudal de 1,2 litros/min. Se alimentará a 12V y consumirá una potencia de 0,9 W. A la salida de la bomba estarán conectadas las tuberías necesarias para llevar el agua a la zona de cultivo. Para la distribución de agua se contarán con pequeños pitorros de 2 l/h. (figura 41).

El control de la bomba se realizará mediante un relé, el cual estará controlado, a su vez, por el microcontrolador. El Arduino será el encargado de activar y desactivar la bomba de riego dependiendo de varios factores:

- La hora configurada por el usuario.
- Si el tanque contiene agua.
- Se realice una prueba de funcionamiento.

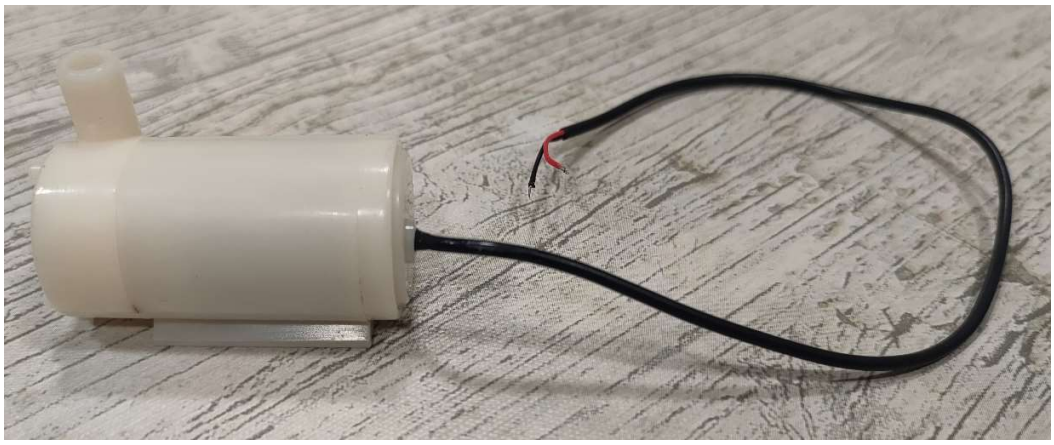


Figura 41. Bomba de agua

6.5.2. Luces LED

Las luces LED se utilizarán para alumbrar el miniinvernadero cuando las horas de sol no sean suficientes para la panta que se esté cultivando. Se colocará una tira de LEDs en el travesaño que cruza toda la estructura, de luz blanca, formada por una tira de 40 cm de LEDs. La tira será alimentada con 12V y consumirá una potencia total de 1.2 W/m.

La propia conexión de la luz contiene un interruptor que permite al usuario encender o apagar. El microcontrolador será el elemento que regule la intensidad de la luz dependiendo de la intensidad de luz que se obtenga a través del luxómetro. La regulación de la intensidad se realizará mediante PWM explicada en el apartado 5.3.

Figura 42. Tira de LEDs

6.5.3. Calefactor

Los sensores de temperatura realizan un seguimiento de la temperatura del habitáculo constante. En ciertas épocas del año, la temperatura puede ser extremadamente baja de tal forma que será necesario compensar esos niveles de temperatura. Se instalará un calefactor en un lateral del invernadero, de modo que no interfiera con la tira de LEDs ni esté cerca de la electrónica que controla todo el sistema.

El calefactor se alimenta con 12V y tiene un consumo de 50W. Será controlado por un relé y el microcontrolador.

El control del calefactor se realizará mediante un relé, el cual estará controlado, a su vez, por el microcontrolador. El Arduino será el encargado de activar y desactivar la bomba de riego dependiendo de varios factores:

- La hora configurada por el usuario.
- Si el tanque contiene agua.
- Se realice una prueba de funcionamiento.

Figura 43. Calefactor

6.6. Sistema de riego

La bomba de agua previamente comentada va a estar conectada a un sistema de riego que permita llegar el agua situada en el tanque a las plantas.

Un sistema de riego por goteo se caracteriza por suministrar pocas cantidades de agua en un tiempo prolongado. Los elementos principales que forman el riego por goteo son: una bomba de agua, una manguera de PVC flexible y goteros.

Las mangueras de PVC se pueden encontrar de distintos diámetros dependiendo de la cantidad de agua que se vaya a transportar. Debido a que no se requiere de una gran cantidad de agua, se dispondrá de un conducto con el diámetro menor posible. Además, se pueden cortar según el usuario prefiera con lo cual, permite un alto nivel de personalización.

Al igual ocurre con los goteros. Existen de varios tamaños y caudales. El caudal se mide en litros por hora (l/h); cuanto mayor sea el caudal, mayor será el gotero y viceversa. Tal y como ocurre con la manguera, no se va a requerir de un caudal elevado con lo que se elegirán los goteros con el mínimo caudal disponible.

Otro motivo por el cual se eligen estos tamaños de los componentes es el tamaño del tanque de almacenamiento de agua y las características de la bomba. La garrafa que actuará como tanque será de entre 6 y 8 litros, como ya se ha comentado anteriormente, de modo que se evitará un caudal grande para evitar que se vacíe rápido y pueda provocar cualquier daño a la bomba. Por otro lado, la conexión de salida de la bomba es de un tamaño reducido ya que es un modelo con una potencia reducida no apta para grandes esfuerzos.

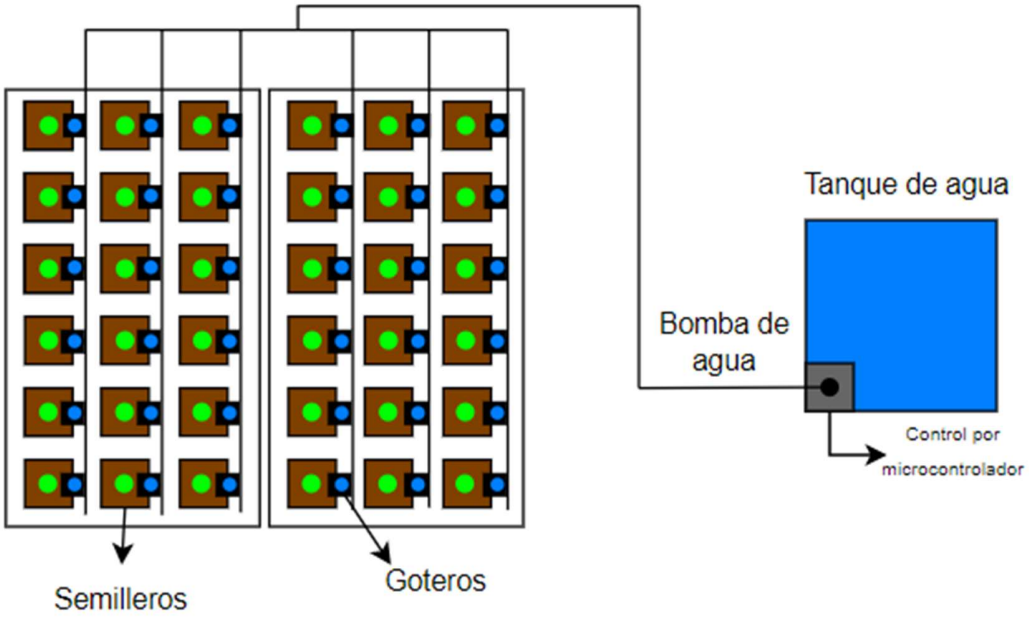


Figura 44. Distribución del sistema de riego planteado

En la figura 44 se observa la distribución que se va a instalar en el invernadero. Los cuadrados marrones con el punto verde representan los semilleros con la respectiva planta. Los cuadrados negros con el punto azul representan los goteros; se van a instalar uno por semillero así cada planta dispondrá de su propio riego. Todos los goteros estarán conectados mediante una manguera a la bomba de agua, representada con el cuadrado gris y el punto negro, dentro del tanque de agua, el cuadrado azul.

6.7. Árbol de potencia

Una vez se han elegido correctamente todos los elementos electrónicos principales, se visualizan en un diagrama clarificando de donde se van a alimentar. Para ello se va a utilizar un árbol de potencia.

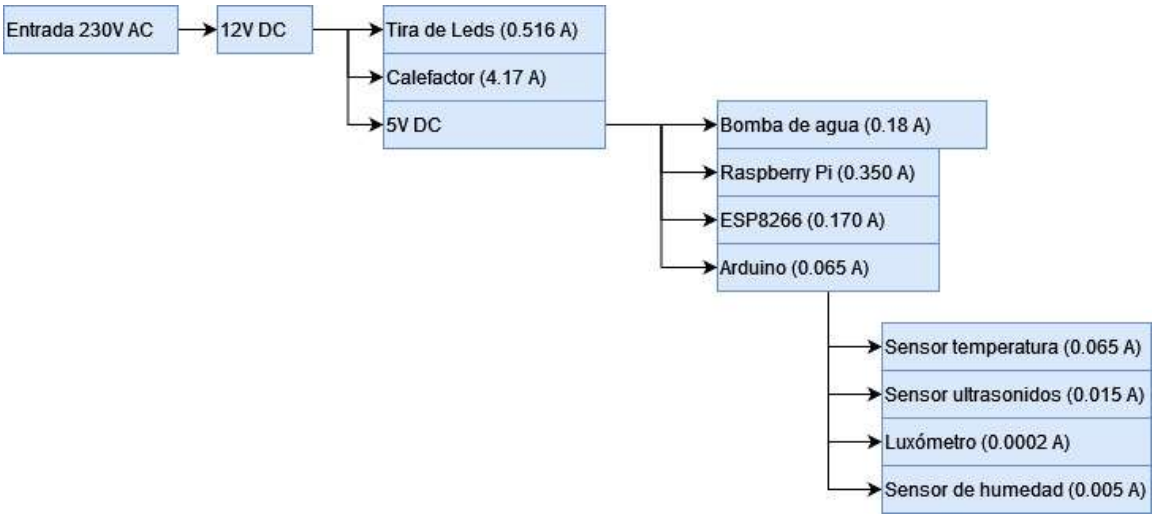


Figura 45. Árbol de potencia

En la imagen anterior se observa que la alimentación principal del sistema de control es la señal de 230V de alterna de la red eléctrica. Se transforma a una señal de continua, en concreto de 12V. No se transforma a una señal de 24 V porque algunos de los elementos necesitan 12V así se conectan directamente. Hay elementos, tales como los microcontroladores (Arduino Mega, modulo ESP8266 y Raspberry Pi) que necesitan 5V para funcionar. Finalmente, los sensores serán alimentados desde el propio Arduino, no será necesaria una conexión externa.

Elemento	Alimentación (V)	Consumo (I)	Consumo (W)
Tira de leds	12 V	0.480 A	5.76 W
Calefactor	12 V	4.170 A	50 W
Arduino	5 V	0.065 A	0.325 W
ESP8266	5 V	0.170 A	0.85 W
Raspberry Pi	5 V	0.380 A	1.9 W
Bomba de agua	5 V	0.600 A	3 W
Sensor temperatura	5 V	0.065 A	0.325 W
Sensor ultrasonidos	5 V	0.015 A	0.075 W
Luxómetro	5 V	0.0002 A	0.001 W
Sensor humedad	5 V	0.005 A	0.025 W

Tabla 2. Tabla de potencia de los equipos del invernadero

6.8. Software

Los datos obtenidos por el sistema van a ser procesados utilizando el Internet de las Cosas (Internet of Things, IoT). (Gracia, s.f.) Este término se refiere a la red por la que distintos dispositivos están conectados entre sí y realizan una comunicación para realizar una o varias funciones. Esta comunicación incluye tanto dispositivos hardware como elementos en la nube. La red puede ser privada, mediante una conexión local, o pública, en este caso, Internet.

Los elementos necesarios para que esta red se pueda llevar a cabo son: un dispositivo que actúe como servidor, es decir, que sea el núcleo central de la red donde se encuentre la

base de datos y los archivos principales para realizar la comunicación servidor-Internet y la comunicación servidor-dispositivos. Es necesaria una conexión a Internet ya sea una red pública o una red local para que la comunicación se lleve a cabo.

El segundo elemento principal es la base de datos. Una herramienta imprescindible para el almacenamiento de datos en la nube permitiendo una gran diversidad de tipos de datos y gran variedad de acciones para el tratamiento de estos.

Y, por último, pero no menos importante, son los dispositivos físicos, es decir, aquellos elementos que se encargan de medir las variables, sensores; realizar acciones, actuadores; y ser el intermediario entre los dos elementos nombrados anteriormente y el servidor, como son los microcontroladores.

Se ha elegido LAMP (Linux Apache MySQL PHP) como plataforma para alojar los sitios web de este proyecto. El dispositivo que va a alojar el servidor es una Raspberry Pi 2. Su sistema operativo es Linux motivo principal por el cual se ha elegido este tipo de servidor. Existe una versión para el sistema operativo Windows llamada WAMP y una versión para macOS llamada MAMP, además de versiones que hábiles para varios sistemas operativos o que utilicen distintas aplicaciones para el servidor o la base de datos.

6.8.1. Linux

Linux es el sistema operativo por excelencia que pertenece a la familia de sistemas operativos tipo Unix de software libre y de código abierto. Linux solo representa el 50% de todo el código del sistema ya que la familia completa incluye desde compiladores hasta entornos de escritorio genéricos o especiales para algún tipo de ámbito como, por ejemplo, la edición de vídeos, programación, etc.

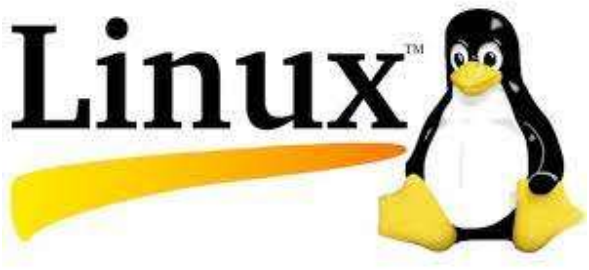


Figura 46. Logotipo del sistema operativo Linux

Al ser una plataforma de código abierto la instalación de aplicaciones y la manejabilidad de todas sus características se ve favorecida con respecto a otros sistemas operativos con menos opciones de configuración.
Otro punto a favor es la opción de no disponer de un entorno de escritorio porque reduce el espacio que el sistema operativo ocupa en la memoria. Las aplicaciones y todas las

opciones de configuración se realizan a través de la consola y mediante comandos que se pueden obtener fácilmente en Internet. Este espacio más amplio en la memoria permite que las aplicaciones necesarias para poner en marcha la plataforma de alojamiento web trabajen más cómodamente sin saturar el procesador de la Raspberry Pi 2. (Wikipedia, 2023)

6.8.2. Apache

El servidor Apache es un servidor web HTTP de código abierto utilizado por cualquier plataforma, Windows, MacOS, pero principalmente por la plataforma Unix en la cual está incluida Linux.

Es un servidor web totalmente gratuito por esto, el 46% de los sitios web de todo el planeta se gestionan utilizando este servidor. Está controlado por los miembros de la empresa Apache Software Foundation ya que una gran comunidad de usuarios puede hacer uso de él.

Figura 47. Logotipo de Apache Software Foundation

La función principal de un servidor web es actuar como intermediario entre el servidor físico y las máquinas clientes, es decir, los dispositivos que se conecten a él y soliciten información. En cada solicitud del usuario la información necesaria es enviada y mostrada en el sitio web. La información puede ser de cualquier tipo: imágenes, texto, video, etc.

Este servidor web es el ideal para aquellos sistemas que no requieran una gran complejidad en sus módulos y no tengan un excesivo flujo de clientes solicitando información porque dispone de gran cantidad de módulos de configuración programables, es sencillo de manejar, multiplataforma, es de código abierto y hay una gran comunidad dando soporte ante cualquier problema y evolucionando el funcionamiento.

Por otro lado, existen problemas de rendimiento que se puedan generar debido a un alto número de solicitudes o los problemas de seguridad debido a sus variados módulos de configuración. (Gustavo, 2023)

6.8.2.1. Proceso de instalación

La instalación de el servidor web Apache es bastante sencilla gracias a la simplicidad de la consola de Linux. Una vez el usuario se ha registrado en la Raspberry, se deberá de introducir los siguientes comandos:

```
sudo apt update
sudo apt install apache2
```

El primer comando sirve para actualizar los paquetes de la Raspberry. En el caso de que el sistema operativo esté recién instalado, no será necesario aplicarlo. El segundo comando es el que se utiliza para instalar los paquetes de Apache. Una vez terminen todas las líneas de la instalación, el servidor estará activo. (Heidi, 2020)

Para comprobar el estado del servidor se utilizará el siguiente comando:

```
sudo systemctl status apache2
```

6.8.3. Base de datos

Una base de datos es un conjunto de datos almacenados siguiendo una estructura predefinida que presentan una relación directa o indirecta entre ellos. Gracias al avance electrónico e informático, las bases de datos se encuentran en formato electrónico facilitando la búsqueda de información y el tratamiento de los datos.

Dos de las bases de datos más conocidas son MySQL y MariaDB. Presentan gran cantidad de similitudes técnicas difíciles de comentar y que ocuparían mucho espacio. Al igual ocurre con sus diferencias, no hay gran cantidad y se pueden resumir en su rendimiento ya que los expertos afirman que MariaDB ofrece mayor rendimiento que MySQL incidiendo en que ambas bases de datos son eficientes en la mayoría de las aplicaciones. (Diana, 2023)

Figura 48. Logotipo de la base de datos MySQL

Figura 49. Logotipo de la base de datos MariaDB

En este proyecto se va a utilizar la base de datos MariaDB ya que, tras una gran cantidad de intentos, la versión de la Raspberry Pi 2 no soportaba la versión de MySQL.

6.8.3.1. Proceso de instalación

Para instalar MariaDB el proceso es el mismo que se ha realizado en el servidor Apache. Se han de introducir los siguientes comandos:

```
sudo apt update
sudo apt install mariadb-server
```

Una vez termine el proceso de instalación, la base de datos estará lista para su uso. El estado de la misma se podrá comprobar con el siguiente comando: (Drake, 2020)

```
sudo systemctl status mariadb
```

6.8.3.2. PhpMyAdmin

PhpMyAdmin es software dedicado a la gestión y administración de las bases de datos basadas en MySQL o MariaDB. Está basado en PHP y permite conectarnos desde servidores remotos para poder tratar todos los datos que almacena.

Existen varias opciones para manejar los datos almacenados en la base de datos. Se puede utilizar el terminal del propio sistema operativo y conectar con la base de datos. En muchos alojamientos tradicionales no está disponible el acceso mediante el terminal por lo que es necesaria una interfaz de usuario que permita el acceso y la gestión de los datos. Estas interfaces pueden estar implementadas bajo aplicaciones de Windows, MacOS o Linux, o mediante aplicaciones web como es el caso de phpMyAdmin.

PhpMyAdmin es la aplicación web más sencilla para administrar las bases de datos basadas en MySQL o MariaDB. Un aspecto importante a tener en cuenta es que se puede acceder a la aplicación web mediante servidores remotos. (García de Zúñiga, 2021)

Este software está basado en el lenguaje de programación PHP de modo que para su utilización basta con subir archivos programados en PHP con el método de acceso a la base de datos correcto y con las funciones que ha de realizar ya sea crear, eliminar o

modificar contenido. En el *Anexo 4 Códigos PHP* se adjuntan los distintos archivos PHP que se han utilizado para actualizar los valores de las bases de datos.

En la figura 48 se observa la interfaz general de phpMyAdmin mientras que en la figura 49 se observa la base de datos creada para este proyecto y las distintas tablas que se van a utilizar.

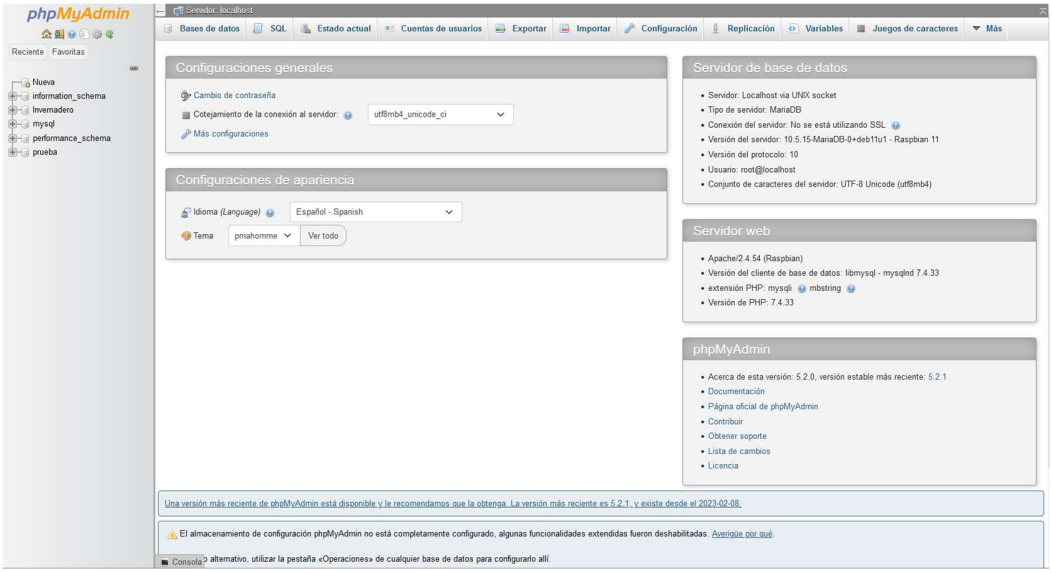


Figura 50. Interfaz principal de phpMyAdmin

	Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
☐	Bomba	Examinar Estructura Buscar Insertar Vaciar Eliminar	2,644	InnoDB	utf8mb4_general_ci	144.0 KB	-
☐	Horarios	Examinar Estructura Buscar Insertar Vaciar Eliminar	35	InnoDB	utf8mb4_general_ci	32.0 KB	-
☐	Humedad	Examinar Estructura Buscar Insertar Vaciar Eliminar	2,629	InnoDB	utf8mb4_general_ci	112.0 KB	-
☐	Luz	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	utf8mb4_general_ci	16.0 KB	-
☐	Nivel	Examinar Estructura Buscar Insertar Vaciar Eliminar	2,765	InnoDB	utf8mb4_general_ci	112.0 KB	-
☐	Temperatura	Examinar Estructura Buscar Insertar Vaciar Eliminar	2,631	InnoDB	utf8mb4_general_ci	112.0 KB	-
	6 tablas	Número de filas	10,704	InnoDB	utf8mb4_general_ci	528.0 KB	0 B

Seleccionar todo Para los elementos que están marcados:

Figura 51. Tablas utilizadas en la base de datos del proyecto

6.8.4. Lenguaje de programación PHP

PHP (*Hypertext Processor*) es un lenguaje de programación de uso general en el entorno del desarrollo web. Es un lenguaje interpretado por parte del servidor y se implementa en mas de 20 millones de aplicaciones web en el mundo.

Este lenguaje se procesa mediante un intérprete PHP implementado como un módulo o un ejecutable. El resultado de este código interpretado y ejecutado suele mostrarse incrustado en una respuesta HTTP. (Wikipedia, 2023)

Pese a que sea un lenguaje cuyo uso principal es el diseño y la gestión de datos de una página web, las aplicaciones de este lenguaje son variadas y extensas como, por ejemplo, tratamiento de la criptografía cuando se usa una Base64, indexación de webs, precocinado de datos, Macrodatos, set de datos, supervisión de datos, set de proposiciones dimensionales, etc.

En este proyecto, el lenguaje PHP va a ser utilizado para el diseño de la página web y el tratamiento de los datos de ésta. Una de las características principales es que este lenguaje puede ser incrustado junto a lenguaje HTML, siguiendo unas ciertas normas de escritura, es por este motivo que la página web principal del sistema combina fragmentos de lenguaje HTML con fragmentos PHP. En el *Anexo 3. Código de la página principal* se puede observar la combinación de ambos lenguajes.

7. Montaje e instalación

La primera parte del montaje se encuentra explicada en el apartado 5.1. Estructura donde ya se especifican las medidas y los pasos a seguir.

Una vez la estructura está montada se dispone a cortar los trozos de plástico para invernadero para cubrir cada una de las zonas abiertas. En las siguientes figuras se muestran un par de imágenes donde se observan algunos de los trozos cortados ya montados.

Figura 52. Parte frontal del miniinvernadero

Figura 53. Parte lateral del miniinvernadero

A continuación, se situarán y se instalarán las placas de los microcontroladores, el módulo de relés y la pequeña placa PCB diseñada dentro de la caja eléctrica. En la figura 54 se muestra la distribución de los elementos nombrados ya instalados.

Figura 54. Caja eléctrica con módulos montados

El siguiente paso es realizar las conexiones de los sensores y actuadores con los microcontroladores. Se ha de destacar que la longitud de los cables varía según la localización de los sensores dentro de la estructura. Visto que este proceso es variable, en la siguiente figura se muestran las conexiones dentro de la caja eléctrica.

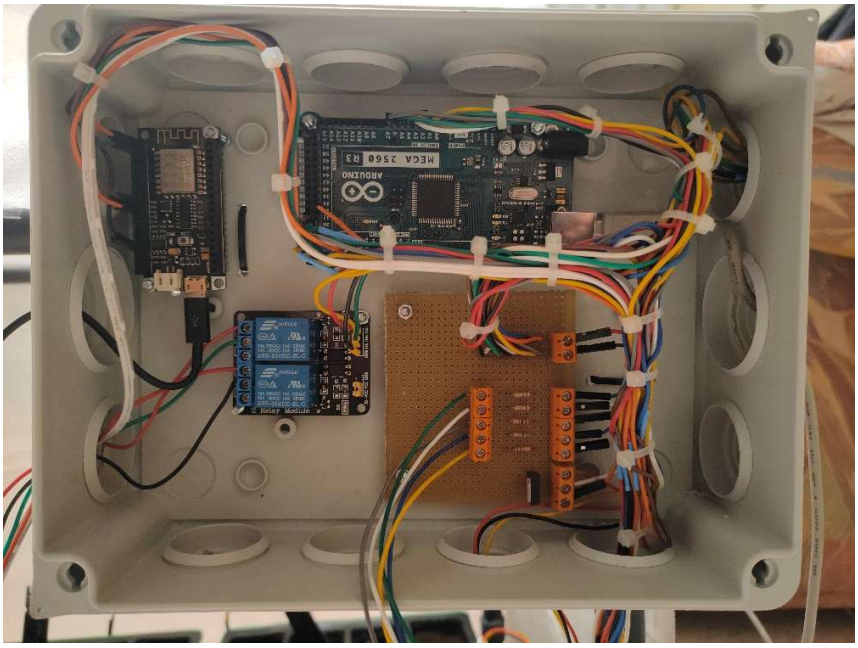


Figura 55. Caja eléctrica con los módulos y las conexiones realizadas

En la tapa de la caja eléctrica se han instalado cinco LEDs y dos interruptores para mostrar el estado de la bomba de agua y del calentador en caso de que la conexión a internet fallase y los interruptores para habilitar las comunicaciones entre placas y activar o desactivar el sistema.

Figura 56. Tapa de la caja eléctrica con interruptores y LEDs

Finalmente, se monta el sistema de riego. Tomando medidas con los semilleros se ajustan las longitudes de las gomas y el número de goteros. Este último valor puede variar dependiendo de la cantidad de agua necesaria para regar o el tamaño del invernadero. En las siguientes figuras se muestra el sistema de riego tanto fuera como dentro del invernadero.

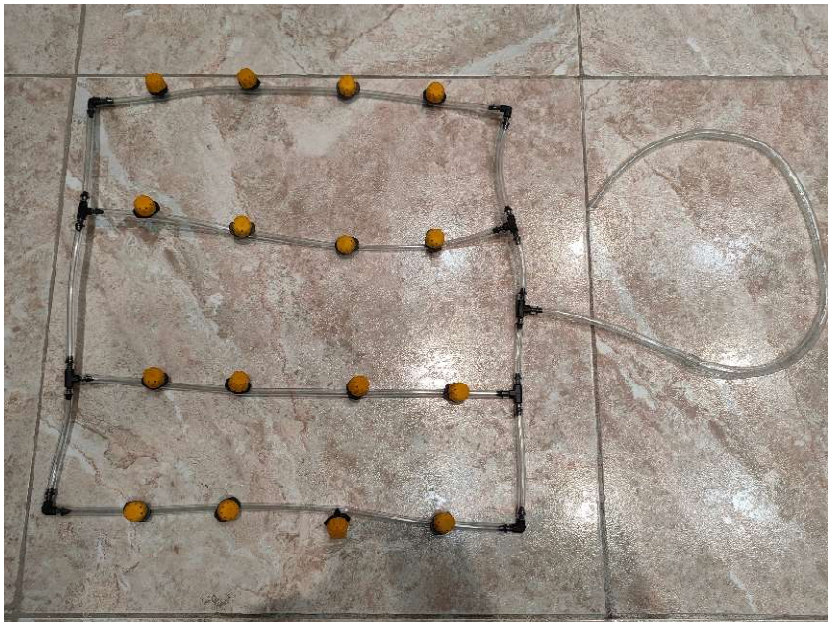


Figura 57. Sistema de riego montado fuera del miniinvernadero

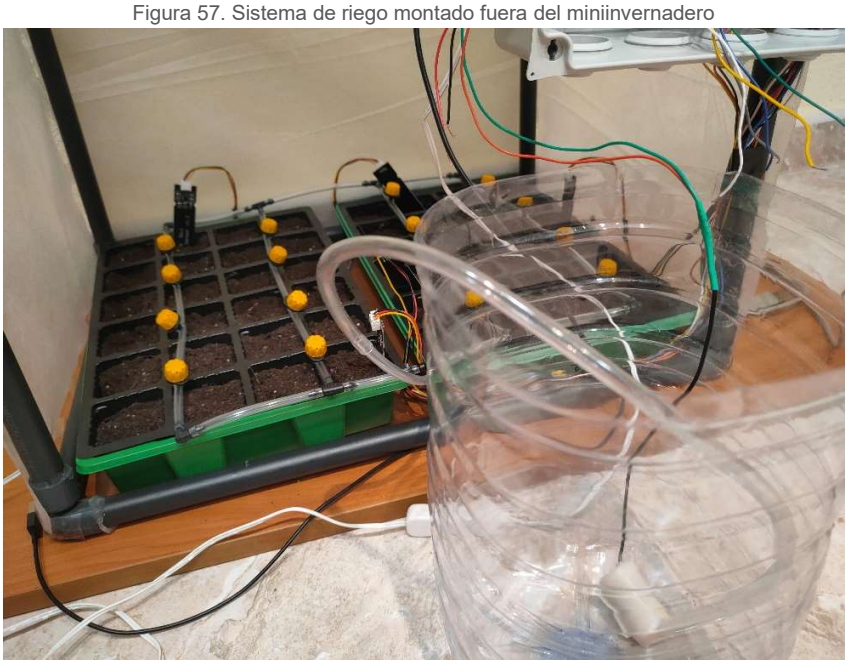


Figura 58. Sistema de riego dentro del invernadero junto con la bomba de agua y el tanque vacío

A continuación, se añaden unas imágenes del resultado final del invernadero listo para ser presentado. Como concepto general, el invernadero ira completamente cerrado teniendo una abertura por la parte frontal para acceder a los semilleros.

Figura 59. Tira de LEDs, calentador y senosr de temperatura instalados dentro del invernadero

Figura 60. Sensores de temperatura instalados dentro de los semilleros

8. Resultados

Se plantaron unas semillas para comprobar el correcto funcionamiento del sistema. Se programaba la hora de regado. Puesto que era verano no se hizo uso del calentador. La s luces LED apenas se encendieron ya que verano es la estación con más horas de luz al día del año. Todos los parámetros fueron bien medidos y al cabo de unas semanas se observó

el crecimiento de las plantas para su trasplantación a un huerto más amplio y así poder terminar correctamente su desarrollo.

Figura 61. Crecimiento de las semillas 1

Figura 62. Crecimiento de las semillas 2

9. Posibles mejoras

Este proyecto está cimentado sobre unos elementos que lo abaratan y, por ende, facilitan el montaje y su uso ya que son más sencillos. Por este motivo, el margen de mejora es muy amplio. Por otro lado, se ha de tener en cuenta que cuanto más se aumente la complejidad de los elementos, más aumentará el precio y se alejará de algunos de los objetivos principales como es el reducido coste del sistema y la simplicidad para usuarios con pocos conocimientos.

Pese a ello, se pueden implementar mejoras que sigan cumpliendo los objetivos de forma correcta. Uno de los aspectos que se pueden mejorar es la interfaz de la página web donde se muestran los datos. Al ser un elemento personalizable dentro del sistema, el usuario

podrá elegir un diseño más básico o complejo. Este diseño más complejo podría incluir gráficas y tablas dinámicas, mostrar los datos de una forma más precisa, programar funciones que puedan automatizar algún proceso, etc. Hay una gran variedad con referencia a este aspecto porque el mundo web ofrece grandes posibilidades a sus usuarios.

El tamaño de la estructura es un punto que también se puede ajustar. Se pueden considerar varias opciones a la hora de configurar el tamaño: ofrecer varios tamaños estándar de modo que el usuario pueda elegir el que más se ajuste a sus necesidades. El coste variará en proporción al tamaño, pero no supondrá un esfuerzo.

El tanque que almacena el agua para el riego es una de las partes que se pueden mejorar ya que ahora mismo se dispone únicamente de una garrafa de 6 litros que puede ser útil para pequeñas demostraciones o cultivos pequeños, pero para un sistema un poco más extenso sería necesario aumentar su tamaño e, incluso, estudiar la posibilidad de cambiar de material a uno más resistente y firme.

10. Conclusiones

El sistema de control que se ha diseñado combina varios aspectos de la electrónica. En primer lugar, la electrónica analógica formada por sensores, actuadores, relés, resistencias, etc. la cual es la primera toma de contacto entre el mundo que nos rodea y la monitorización de datos. En segundo lugar, la programación; tanto la programación de los microcontroladores, que se encargan de recibir, procesar y transmitir los datos; y la programación web, la encargada de mostrar y dar formato a los datos tratados y enviados por los microcontroladores. En ambos casos, los valores obtenidos pueden ser almacenados ya sea en unas variables o en una base de datos.

No solo se ha quedado en un diseño simulado o en un estudio, sino que el sistema se ha llevado a cabo y se ha podido montar comprobando que las soluciones planteadas funcionan correctamente y tienen una utilidad dentro del mundo de los invernaderos.

Por último, nombrar la motivación principal y personal para llevar a cabo el proyecto. Siempre he estado interesado en el mundo del cultivo ya que desde pequeño he estado rodeado de siembras locales para consumo propio, pero siempre partiendo de un brote previamente germinado en viveros o tiendas especializadas. Poder realizar este sistema y ver que funciona inspira a poder realizar el proceso de cultivo completo, desde la germinación hasta la recolección controlando desde un principio el desarrollo de las plantas.

11. Bibliografía

78L10 Datasheet. (s.f.). Obtenido de https://www.alldatasheet.com/view.jsp?Searchword=78l10%20datasheet&gclid=Cj0KCQiAtICdBhCLARIsALUBFcGXEgbQCZ9ou8Jltfe6xSvxLNSbMYNDESEaq6_ftkw0_Lv8x1baxwUaAg2lEALw_wcB

AD620 Datasheet. (s.f.). Obtenido de https://www.alldatasheet.com/view.jsp?Searchword=Ad620%20datasheet&gclid=Cj0KCQiAtICdBhCLARIsALUBFcHp6FNrvtpYR3lY1Jl4_zL0KtS4ViBIhkC1VUW7Vl3h1e0kmLnFHGcaAr5TEALw_wcB

Agropinos. (07 de Octubre de 2021). Obtenido de https://www.agropinos.com/blog/historia-del-invernadero-para-cultivos

Amazon. (s.f.). Obtenido de https://www.amazon.es/AZDelivery-NodeMCU-Amica-Modul-Parent/dp/B082DJVXFC

Amplificador Restador. (s.f.). Obtenido de https://solucioningenieril.com/amplificadores_operacionales/amplificador_restador

Arrow. (22 de Diciembre de 2017). Obtenido de https://www.arrow.com/es-mx/research-and-events/articles/arduino-uno-vs-mega-vs-micro

Asoven. (12 de Septiembre de 2018). *Asoven*. Obtenido de https://www.asoven.com/pvc/que-es-el-pvc-ventajas-fabricacion-e-impacto-ambiental/

BricoGeek. (s.f.). Obtenido de https://tienda.bricogeek.com/arduino/306-arduino-mega-2560.html

BS250P Datasheet. (s.f.). Obtenido de https://www.alldatasheet.es/datasheet-pdf/pdf/95978/GE/BS250.html

Debian User Forum. (2013). Obtenido de https://forums.debian.net/viewtopic.php?t=151504

del Valle Hernández, L. (2018). *ProgramarFacil*. Obtenido de https://programarfacil.com/podcast/esp8266-wifi-coste-arduino/

Diana. (8 de Febrero de 2023). *Hostinger*. Obtenido de https://www.hostinger.es/tutoriales/mariadb-vs-mysql

Drake, M. (11 de Junio de 2020). *DigitalOcean*. Obtenido de https://www.digitalocean.com/community/tutorials/how-to-install-mariadb-on-ubuntu-20-04-es

Farnell. (s.f.). Obtenido de https://es.farnell.com/vishay/bpw77na/phototransistor-npn-3-to-18/dp/1612659

Feria de tecnologia. (s.f.). Obtenido de https://www.feriadetecnologia.com/arduino/informacin_ldr.html

Fernández, Y. (23 de Septiembre de 2022). *Xataka*. Obtenido de https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno

García de Zúñiga, F. (25 de 11 de 2021). *Arsys*. Obtenido de https://www.arsys.es/blog/phpmyadmin

GitHub. (2020). Obtenido de https://gist.github.com/TrickSumo/37909670934bc1b691fca35accf855c9

Gracia, M. (s.f.). *Deloitte*. Obtenido de https://www2.deloitte.com/es/es/pages/technology/articles/IoT-internet-of-things.html

Gustavo, B. (24 de Mayo de 2023). *Hostinger*. Obtenido de https://www.hostinger.es/tutoriales/que-es-apache/

Heidi, E. (19 de Mayo de 2020). *DigitalOcean*. Obtenido de https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-20-04-es

Inoxidables VIctoria. (1 de Septiembre de 2022). Obtenido de https://inoxidablesvictoria.com/blog/ventajas-desventajas-aluminio/

IoT Project Ideas. (20 de Junio de 2020). Obtenido de https://iotprojectsideas.com/insert-data-into-mysql-database-with-esp8266/

IoTicos. (2019). *YouTube*. Obtenido de https://www.youtube.com/watch?v=uLkpILpQKuY

Mecatrónica LATAM. (5 de Mayo de 2021). Obtenido de https://www.mecatronicalatam.com/es/tutoriales/sensores/sensor-de-luz/

Menna. (s.f.). *Como Funciona*. Obtenido de https://como-funciona.co/fotodiodo/?utm_content=cmp-true

Naciones Unidas. (s.f.). Obtenido de https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/

Naylamp Mechatronics. (2016). Obtenido de https://naylampmechatronics.com/blog/56_usando-esp8266-con-el-ide-de-arduino.html

Ouyang, A. (11 de Septiembre de 2022). *Seeedstudio*. Obtenido de https://www.seeedstudio.com/blog/2022/07/22/%EF%BF%BCdifferent-types-of-soil-moisture-sensors%EF%BF%BC/

Parzibyte's blog. (23 de Noviembre de 2020). Obtenido de https://parzibyte.me/blog/2020/11/23/peticion-http-esp8266/

PNGWing. (s.f.). Obtenido de https://www.pngwing.com/es

Portela Rincon, N. A. (Enero de 2023). *Youtube*. Obtenido de https://www.youtube.com/watch?v=84y-qRFouK4

Restuccia, R. (11 de Marzo de 2021). *Jainsusa*. Obtenido de https://jainsusa.com/blog/quick-guide-soil-moisture-sensors/

Santos, S. (2020). *Random Nerd Tutorials*. Obtenido de https://randomnerdtutorials.com/esp32-http-get-post-arduino/

SensaCultivo. (6 de Junio de 2022). Obtenido de https://sensacultivo.es/2022/06/06/sondas-de-humedad-en-suelo-tipos-y-caracteristicas-principales/

Shoptronica. (s.f.). Obtenido de https://www.shoptronica.com/524-curiosidades-tutoriales-y-gadgets

Soloelectronicos. (11 de Agosto de 2022). *SoloElectronicos*. Obtenido de https://soloelectronicos.com/2022/08/11/obtener-la-fecha-y-hora-del-servidor-ntp-con-esp32/?utm_content=cmp-true

Tech, T. T. (2021). *YouTube*. Obtenido de https://www.youtube.com/watch?v=baYfrxay8zM

Texolab. (25 de Octubre de 2018). Obtenido de https://texolab.net/2018/10/25/sincronizar-la-hora-en-esp8266-con-servidor-ntp/

Wikipedia. (23 de Agosto de 2023). Obtenido de https://es.wikipedia.org/wiki/PHP

Wikipedia. (11 de Septiembre de 2023). Obtenido de https://es.wikipedia.org/wiki/GNU/Linux

Wikipedia. (13 de Marzo de 2023). Obtenido de https://es.wikipedia.org/wiki/Sensor_de_nivel

Wikipedia. (18 de Julio de 2023). Obtenido de https://es.wikipedia.org/wiki/Microcontrolador#Historia

XTR110 Datasheet. (s.f.). Obtenido de http://www.datasheet.es/PDF/285254/XTR110-pdf.html

Anexo 1. Código del Arduino Mega

```
#include <BH1750.h>
#include <Wire.h>
//Pines de transmision
#define RX0 0
#define TX0 1

#define RX1 19
#define TX1 18

#define RX2 17
#define TX2 16

#define RX3 15
#define TX3 14

#define SDA 20
#define SCL 21

#define tempIn1 A0
#define tempIn2 A1
#define moistureIn1 A2
#define moistureIn2 A3
#define moistureIn3 A4
#define moistureIn4 A5
#define moistureIn5 A6
#define echoPin 22
#define triggerPin 23
#define pumpActLed 24
#define pumpAct 25
#define pumpReadyLed 26
#define fillTankLed 27
#define heaterOnLed 28
#define heaterAct 29
#define heaterOffLed 30
#define pStatus 31
#define hStatus 32
#define stopAll 33
#define lightAct 34

//Variables
float tempValue1, tempValue2, tempTotal;
float   moisValue1,   moisValue2,   moisValue3,   moisValue4,   moisValue5,
moisTotal;
float levelValue, lightValue;
char action;
const float velSon = 34000.0;
int tState, pReady, pAct=0, hAct=0;
int reading, ledValue;

BH1750 luxometro;

void setup() {
  Wire.begin();
```

```
  Serial.begin(115200);
  Serial1.begin(115200);
  luxometro.begin(BH1750::CONTINUOUS_HIGH_RES_MODE, 0x23);

  pinMode(tempIn1, INPUT);
  pinMode(tempIn2, INPUT);
  pinMode(moistureIn1, INPUT);
  pinMode(moistureIn2, INPUT);
  pinMode(moistureIn3, INPUT);
  pinMode(moistureIn4, INPUT);
  pinMode(moistureIn5, INPUT);
  pinMode(echoPin, INPUT);
  pinMode(pStatus, INPUT);
  pinMode(hStatus, INPUT);
  pinMode(stopAll, INPUT);

  pinMode(triggerPin, OUTPUT);
  pinMode(pumpActLed, OUTPUT);
  pinMode(pumpAct, OUTPUT);
  pinMode(fillTankLed, OUTPUT);
  pinMode(heaterAct, OUTPUT);
  pinMode(lightAct, OUTPUT);
  pinMode(pumpReadyLed, OUTPUT);
  pinMode(heaterOnLed, OUTPUT);
  pinMode(heaterOffLed, OUTPUT);
}

void loop() {
  if(digitalRead(stopAll)==0){
    triggerInit();
    levelValue=levelMeasure();
    sendLevel(levelValue);
    delay(2000);
    tempValue1=tempMeasure(tempIn1);
    tempValue2=tempMeasure(tempIn2);
    tempTotal=meanTemperature(tempValue1, tempValue2);
    sendTemp(tempTotal);
    delay(2000);
    moisValue1=moisMeasure(moistureIn1);
    moisValue2=moisMeasure(moistureIn2);
    moisValue3=moisMeasure(moistureIn3);
    moisValue4=moisMeasure(moistureIn4);
    moisValue5=moisMeasure(moistureIn5);

moisTotal=meanMoisture(moisValue1,moisValue2,moisValue3,moisValue4,moisVa
lue5);
    sendMoisture(moisTotal);
    delay(2000);
    lightValue=lightMeasure();
    sendLuz(lightValue);
    ledValue=255-(lightValue*255/28067);
    showValues(levelValue, tempValue1, tempValue2, tempTotal, moisValue1,
moisValue2, moisValue3, moisValue4, moisValue5, moisTotal, lightValue);
    pAct=digitalRead(pStatus);
    hAct=digitalRead(hStatus);
    Serial.println(pAct);
```

```
    Serial.println(hAct);
    delay(5000);
    if(tempTotal<=5.00 || hAct == 1){
      digitalWrite(heaterAct, HIGH);
      digitalWrite(heaterOnLed, HIGH);
      digitalWrite(heaterOffLed, LOW);
    }else{
      digitalWrite(heaterAct, LOW);
      digitalWrite(heaterOnLed, LOW);
      digitalWrite(heaterOffLed, HIGH);
    }
    if(levelValue>=25.00){
      emptyTank();
    }else{
      tankEnabled();
    }

    if(moisTotal<=35){
      pAct=1; //Activar la bomba
      digitalWrite(pumpAct, HIGH);
    }
    sendPump(tState, pReady, pAct);
    if(pAct==1 && pReady==1){
      digitalWrite(pumpAct, HIGH);
      digitalWrite(pumpActLed, HIGH);
      Serial.println("Estoy regando");
    }else{
      digitalWrite(pumpAct, LOW);
      digitalWrite(pumpActLed, LOW);
      Serial.println("No estoy regando");
    }
    delay(2000);//Se comprueba el estado del tanque por si se queda sin
agua mientras riega
    triggerInit();
    levelValue=levelMeasure();
    sendLevel(levelValue);
    if(levelValue>=25.00){ //Cuando se instale el sensor en la garrafa,
modificar la distancia
      emptyTank();
    }else{
      tankEnabled();
    }
    sendPump(tState, pReady, pAct, hAct);
    if(lightValue<=250){
      digitalWrite(lightAct, HIGH);
    }
  }else if(digitalRead(stopAll)==1){
    Serial.println("TODO STOP");
  }
}

void triggerInit(){
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(10);
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
```

```
  digitalWrite(triggerPin, LOW);
}

float levelMeasure(void){
  unsigned long time = pulseIn(echoPin, HIGH);
  float result = time*0.000001*velSon/2.0;
  return result;
}

float tempMeasure(uint8_t pin){
  float result=((analogRead(pin)*(5.0/1024.0))-0.5)/0.01;
  return result;
}

float meanTemperature(float temp1, float temp2){
  float result = (temp1+temp2)/2.0;
  return result;
}

float moisMeasure(uint8_t pin){
  float result = analogRead(pin)*5.0/1023.0;
  return result;
}

float meanMoisture(float mois1, float mois2, float mois3, float mois4,
float mois5){
  float result = 100.0-((mois1+mois2+mois3+mois4+mois5)/5.0*(100.0/5.0));
  return result;
}

float lightMeasure(void){
  float result = luxometro.readLightLevel();
  return result;
}

void emptyTank(void){
  Serial.println("Rellenar tanque");
  tState=1;
  pReady=0;
  pAct=0;
  digitalWrite(pumpActLed, LOW);
  digitalWrite(pumpReadyLed, LOW);
  digitalWrite(fillTankLed, HIGH);
}

void tankEnabled(void){
 Serial.println("Tanque OK");
  tState=0;
  digitalWrite(fillTankLed, LOW);
  pReady=1;
  digitalWrite(pumpReadyLed, HIGH);
}

void showValues(float level, float temp1, float temp2, float temp, float
mois1, float mois2, float mois3, float mois4, float mois5, float mois,
float light){
```

```
  Serial.print("Level Value: ");
  Serial.println(level);
  Serial.print("Temperature: ");
  Serial.print(temp1);
  Serial.print(" ");
  Serial.print(temp2);
  Serial.print(" ");
  Serial.println(temp);
  Serial.print("Moisture: ");
  Serial.print(mois1);
  Serial.print(" ");
  Serial.print(mois2);
  Serial.print(" ");
  Serial.print(mois3);
  Serial.print(" ");
  Serial.print(mois4);
  Serial.print(" ");
  Serial.print(mois5);
  Serial.print(" ");
  Serial.println(mois);
  Serial.print("Light Value: ");
  Serial.println(light);
}

void sendTemp(float temp){
  int value1, value2, value3, value4;
  float num=temp;
  Serial1.write('T');
  value1 = temp/10;
  Serial1.write(value1);
  delay(200);
  value2 = num-(value1*10);
  Serial1.write(value2);
  delay(200);
  num = num-(value1*10)-(value2);
  value3 = num/0.1;
  Serial1.write(value3);
  delay(200);
  num = num-(value3*0.1);
  value4 = num/0.01;
  Serial1.write(value4);
  delay(200);
  /*Serial.print(value1);
  Serial.print(value2);
  Serial.print(value3);
  Serial.println(value4);*/
}

void sendMoisture(float moisture){
  int value1, value2, value3, value4;
  float num=moisture;
  Serial1.write('H');
  value1 = moisture/10;
  Serial1.write(value1);
  delay(200);
  value2 = num-(value1*10);
```

```
  Serial1.write(value2);
  delay(200);
  num = num-(value1*10)-(value2);
  value3 = num/0.1;
  Serial1.write(value3);
  delay(200);
  num = num-(value3*0.1);
  value4 = num/0.01;
  Serial1.write(value4);
  delay(200);
  /*Serial.print(value1);
  Serial.print(value2);
  Serial.print(value3);
  Serial.println(value4);*/
}

void sendLevel(float level){
  int value1, value2, value3, value4;
  float num=level;
  Serial1.write('W');
  value1 = level/10;
  Serial1.write(value1);
  delay(200);
  value2 = num-(value1*10);
  Serial1.write(value2);
  delay(200);
  num = num-(value1*10)-(value2);
  value3 = num/0.1;
  Serial1.write(value3);
  delay(200);
  num = num-(value3*0.1);
  value4 = num/0.01;
  Serial1.write(value4);
  delay(200);
  /*Serial.print(value1);
  Serial.print(value2);
  Serial.print(value3);
  Serial.println(value4);*/
}

void sendLuz(uint16_t luz){
  int valor1, valor2, valor3, valor4, valor5;
  Serial1.write('L');
  valor1=luz/10000;
  Serial1.write(valor1);
  delay(200);
  valor2=(luz-(valor1*10000))/1000;
  Serial1.write(valor2);
  delay(200);
  valor3=(luz-(valor1*10000)-(valor2*1000))/100;
  Serial1.write(valor3);
  delay(200);
  valor4=(luz-(valor1*10000)-(valor2*1000)-(valor3*100))/10;
  Serial1.write(valor4);
  delay(200);
  valor5=luz-(valor1*10000)-(valor2*1000)-(valor3*100)-(valor4*10);
```

```
  Serial1.write(valor5);
  delay(200);
}

void sendPump(int tankState, int pumpPreparation, int pumpState, int
heaterState){
  Serial1.write('P');
  delay(200);
  Serial.print("Tank State: ");
  Serial.println(tankState);
  Serial1.write(tankState);
  delay(200);
  Serial.print("Pump ready: ");
  Serial.println(pumpPreparation);
  Serial1.write(pumpPreparation);
  delay(200);
  Serial.print("Pump State: ");
  Serial.println(pumpState);
  Serial1.write(pumpState);
  delay(200);
  Serial.print("Heater State: ");
  Serial.println(heaterState);
  Serial1.write(heaterState);
  delay(200);
}
```

Anexo 2. Código del módulo ESP8266

```
#include <ESP8266WiFiMulti.h>
#include <ESP8266HTTPClient.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
#include <SoftwareSerial.h>

const char* ssid = "Comjazz4";
const char* password = "1pi2so3pi4so";
const char* host = "172.16.2.131";
char sensor;
float tempValue, moisValue, levelValue;
int lightValue;
int tState, pReady, pAct, pTime, hTime, pumpTimeAct=0, heaterTimeAct=0,
hAct;
int pHour, pMin, pDuration, hHour, hMin, hDuration, interval;
String zero="0", pInitTime, pOffTime, hInitTime, hOffTime;

WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", 7200, 5000);
//SoftwareSerial espSerial(1,3); // RX(D3), TX(D1)

void setup() {
  Serial.begin(115200);
  Serial1.begin(115200);
  pinMode(D6, OUTPUT);
  WiFi.begin(ssid, password);
  Serial.print("Conectando...");
  while (WiFi.status() != WL_CONNECTED) { //Check for the connection
    delay(500);
    Serial.print(".");
  }
  Serial.print("Conectado con éxito, mi IP es: ");
  Serial.println(WiFi.localIP());
  timeClient.begin();
}

void loop() {
// T= Sensor de temperatura; H = Sensor de humedad; W = Sensor de nivel; L
= Luxómetro;
  if(WiFi.status()==WL_CONNECTED){
    HTTPClient http;
    WiFiClient client;
    sensor = Serial.read();
    if(sensor == 'T'){
      tempValue=recieveTemp();
      Serial.print("Temperatura: ");
      Serial.println(tempValue);
      uploadTemp(tempValue);
    }else if(sensor == 'H'){
      moisValue=recieveMoisture();
      Serial.print("Humedad: ");
      Serial.println(moisValue);
      uploadMois(moisValue);
```

```
    }else if(sensor == 'W'){
      levelValue=recieveLevel();
      Serial.print("Nivel: ");
      Serial.println(levelValue);
      uploadLevel(levelValue);
    }else if(sensor == 'L'){
      lightValue=recieveLux();
      Serial.print("Luz: ");
      Serial.println(lightValue);
      uploadLight(lightValue);
    }else if(sensor == 'P'){
      recievePump();
      Serial.print("Tank state: ");
      Serial.println(tState);
      Serial.print("Pump preparation: ");
      Serial.println(pReady);
      Serial.print("Pump state ");
      Serial.println(pAct);
      uploadPump(tState, pReady, pAct, hAct);
    }
    pTime=esp_recievePumpTime();
    hTime=esp_recieveHeatTime();
    pumpTime(pTime);
    heatTime(hTime);
    delay(2000);
    pInitTime=initialTime(pHour, pMin);
    pOffTime=finalTime(pHour, pMin, pDuration);
    hInitTime=initialTime(hHour, hMin);
    hOffTime=finalTime(hHour, hMin, hDuration);
    Serial.print("Inicio bomba: ");
    Serial.println(pInitTime);
    Serial.print("Fin bomba: ");
    Serial.println(pOffTime);
    Serial.print("Inicio calefactor: ");
    Serial.println(hInitTime);
    Serial.print("Fin calefactor: ");
    Serial.println(hOffTime);
    pumpActivation(pInitTime, pOffTime, pAct, pumpTimeAct);
    heaterActivation(hInitTime, hOffTime, hAct, heaterTimeAct);
  }
}
void  pumpActivation(String  init,  String  final,  int  pumpState,  int
timePump){
  timeClient.update();
  Serial.println(timeClient.getFormattedTime());
  if(timeClient.getFormattedTime().startsWith(init) && timePump == 0){
    Serial.println("Riego activado");
    pumpTimeAct=1;
    pAct=1;
    digitalWrite(D6, HIGH);
  }else if(timePump == 1){
    if(timeClient.getFormattedTime().startsWith(final)){
      Serial.println("Riego terminado");
      pumpTimeAct=0;
      pAct=0;
      digitalWrite(D6, LOW);
```

```
    }else{
      Serial.println("Sigo regando...");
      pAct=1;
      digitalWrite(D6, HIGH);
    }
  }else{
    Serial.println("No es hora");
    pAct=0;
    pumpTimeAct=0;
    digitalWrite(D6, LOW);
  }
}
void heaterActivation(String init, String final, int heaterState, int
timeHeater){
  timeClient.update();
  Serial.println(timeClient.getFormattedTime());
  if(timeClient.getFormattedTime().startsWith(init) && timeHeater == 0){
    Serial.println("Riego activado");
    heaterTimeAct=1;
    hAct=1;
    digitalWrite(D5, HIGH);
  }else if(timeHeater == 1){
    if(timeClient.getFormattedTime().startsWith(final)){
      Serial.println("Riego terminado");
      heaterTimeAct=0;
      hAct=0;
      digitalWrite(D5, LOW);
    }else{
      Serial.println("Sigo regando...");
      hAct=1;
      digitalWrite(D5, HIGH);
    }
  }else{
    Serial.println("No es hora");
    hAct=0;
    heaterTimeAct=0;
    digitalWrite(D5, LOW);
  }
}
String initialTime(int hour, int minutes){
  String initTime, tHour, tMin;
  tHour = String(hour);
  tMin = String(minutes);
  initTime = tHour + ":" + tMin;
  if(hour<10){
    initTime = zero + initTime;
  }
  if(hour==24){
    initTime = zero + zero + ":" + tMin;
  }
  if(minutes<10){
    String part1 = initTime.substring(0,3);
    String part2 = initTime.substring(3);
    initTime = part1 + zero + part2;
  }
  return initTime;
```

```
}
String finalTime(int hour, int minutes, int duration){
  String offTime;
  int offHour, offMin;
  if(minutes+duration>=60 && minutes+duration<120){
    offHour = hour+1;
    offMin = (minutes+duration)%60;
    if(offHour>24){
      offHour = offHour - 24;
    }
    offTime = String(offHour) + ":" + String(offMin);
    if(offHour<10){
      offTime = zero + offTime;
    }
    if(offHour==24){
      offTime = zero + zero + ":" + String(offMin);
    }
    if(offMin<10){
      String part1 = offTime.substring(0,3);
      String part2 = offTime.substring(3);
      offTime = part1 + zero + part2;
    }
  }else if(minutes+duration>=120){
    offHour = hour+2;
    offMin = (minutes+duration)%60;
    if(offHour>24){
      offHour = offHour - 24;
    }
    offTime = String(offHour) + ":" + String(offMin);
    if(offHour<10){
      offTime = zero + offTime;
    }
    if(offHour==24){
      offTime = zero + zero + ":" + String(offMin);
    }
    if(offMin<10){
      String part1 = offTime.substring(0,3);
      String part2 = offTime.substring(3);
      offTime = part1 + zero + part2;
    }
  }else if(minutes+duration<60){
    offHour = hour;
    offMin = (minutes+duration)%60;
    if(offHour>24){
      offHour = offHour - 24;
    }
    offTime = String(offHour) + ":" + String(offMin);
    if(offHour<10){
      offTime = zero + offTime;
    }
    if(offHour==24){
      offTime = zero + zero + ":" + String(offMin);
    }
    if(offMin<10){
      String part1 = offTime.substring(0,3);
      String part2 = offTime.substring(3);
```

```
      offTime = part1 + zero + part2;
    }
  }
  return offTime;
}
void pumpTime(int time){
  //Formato que se recibe la hora XXYYZZ
  //XX->Minutos de inicio; YY->Duracion; ZZ->Hora inicio
  String initTime;
  if(time%100>=10 && time%100<=24){
    pHour=time%100;
    interval=(time-pHour)/100;
  }else{
    pHour=time%10;
    interval=(time-pHour)/10;
  }
  pDuration=interval%100;
  pMin=(interval-pDuration)/100;
}
void heatTime(int time){
  //Formato que se recibe la hora XXYYZZ
  //XX->Minutos de inicio; YY->Duracion; ZZ->Hora inicio
  String initTime;
  if(time%100>=10 && time%100<=24){
    hHour=time%100;
    interval=(time-hHour)/100;
  }else{
    hHour=time%10;
    interval=(time-hHour)/10;
  }
  hDuration=interval%100;
  hMin=(interval-hDuration)/100;
}
float recieveTemp(void){
  float num;
  int value1 = Serial.read();
  delay(200);
  int value2 = Serial.read();
  delay(200);
  int value3 = Serial.read();
  delay(200);
  int value4 = Serial.read();
  delay(200);
  num = (value1*10)+value2+value3*0.1+value4*0.01;
  return num;
}
float recieveMoisture(void){
  float num;
  int value1 = Serial.read();
  delay(200);
  int value2 = Serial.read();
  delay(200);
  int value3 = Serial.read();
  delay(200);
  int value4 = Serial.read();
  delay(200);
```

```
  num = (value1*10)+value2+value3*0.1+value4*0.01;
  return num;
}
float recieveLevel(void){
  float num;
  int value1 = Serial.read();
  delay(200);
  int value2 = Serial.read();
  delay(200);
  int value3 = Serial.read();
  delay(200);
  int value4 = Serial.read();
  delay(200);
  num = (value1*10)+value2+value3*0.1+value4*0.01;
  return num;
}
int recieveLux(void){
  int num;
  int value1 = Serial.read();
  delay(200);
  int value2 = Serial.read();
  delay(200);
  int value3 = Serial.read();
  delay(200);
  int value4 = Serial.read();
  delay(200);
  int value5 = Serial.read();
  delay(200);
  num = (value1*10000)+(value2*1000)+(value3*100)+(value4*10)+(value5);
  return num;
}
void recievePump(void){
  tState = Serial.read();
  delay(200);
  pReady = Serial.read();
  delay(200);
  pAct = Serial.read();
  delay(200);
  hAct = Serial.read();
  delay(200);
}
void uploadTemp(float temp){
  HTTPClient http;
  WiFiClient client;
  String temperatura = String(temp);
  String datos_a_enviar = "temp=" + temperatura;
  http.begin(client,                  "http://172.16.2.131/postTemp.php");
//Indicamos el destino
  http.addHeader("Content-Type",     "application/x-www-form-urlencoded");
//Preparamos el header text/plain si solo vamos a enviar texto plano sin
un paradigma llave:valor.
  float codigo_respuesta = http.POST(datos_a_enviar);   //Enviamos el post
pasándole, los datos que queremos enviar. (esta función nos devuelve un
código que guardamos en un int)
  if(codigo_respuesta>0){
    //Serial.println("Código HTTP ► " );   //Print return code
```

```
    //Serial.println(codigo_respuesta);
    if(codigo_respuesta == 200){
      String cuerpo_respuesta = http.getString();
      Serial.print("El servidor respondió: ");
      Serial.println(cuerpo_respuesta);
      }
  }else{
    Serial.print("Error enviando POST, código: ");
    Serial.println(codigo_respuesta);
  }
  http.end();  //libero recursos
}
void uploadMois(float mois){
  HTTPClient http;
  WiFiClient client;
  String humedad = String(mois);
  String datos_a_enviar = "mois=" + humedad;
  http.begin(client,                  "http://172.16.2.131/postMois.php");
//Indicamos el destino
  http.addHeader("Content-Type",    "application/x-www-form-urlencoded");
//Preparamos el header text/plain si solo vamos a enviar texto plano sin
un paradigma llave:valor.
  float codigo_respuesta = http.POST(datos_a_enviar);   //Enviamos el post
pasándole, los datos que queremos enviar. (esta función nos devuelve un
código que guardamos en un int)
  if(codigo_respuesta>0){
    //Serial.println("Código HTTP ► " );   //Print return code
    //Serial.println(codigo_respuesta);
    if(codigo_respuesta == 200){
      String cuerpo_respuesta = http.getString();
      Serial.print("El servidor respondió: ");
      Serial.println(cuerpo_respuesta);
      }
  }else{
    Serial.print("Error enviando POST, código: ");
    Serial.println(codigo_respuesta);
  }
  http.end();  //libero recursos
}
void uploadLevel(float level){
  HTTPClient http;
  WiFiClient client;
  String nivel = String(level);
  String datos_a_enviar = "level=" + nivel;
  http.begin(client,                "http://172.16.2.131/postLevel.php");
//Indicamos el destino
  http.addHeader("Content-Type",    "application/x-www-form-urlencoded");
//Preparamos el header text/plain si solo vamos a enviar texto plano sin
un paradigma llave:valor.
  float codigo_respuesta = http.POST(datos_a_enviar);   //Enviamos el post
pasándole, los datos que queremos enviar. (esta función nos devuelve un
código que guardamos en un int)
  if(codigo_respuesta>0){
    //Serial.println("Código HTTP ► " );   //Print return code
    //Serial.println(codigo_respuesta);
    if(codigo_respuesta == 200){
```

```
      String cuerpo_respuesta = http.getString();
      Serial.print("El servidor respondió: ");
      Serial.println(cuerpo_respuesta);
      }
  }else{
    Serial.print("Error enviando POST, código: ");
    Serial.println(codigo_respuesta);
  }
  http.end();  //libero recursos
}
void uploadLight(int light){
  char datos_a_enviar = "light=" + light;
  http.begin(client,                "http://172.16.2.131/postLight.php");
//Indicamos el destino
  http.addHeader("Content-Type",    "application/x-www-form-urlencoded");
//Preparamos el header text/plain si solo vamos a enviar texto plano sin
un paradigma llave:valor.
  float codigo_respuesta = http.POST(datos_a_enviar);   //Enviamos el post
pasándole, los datos que queremos enviar. (esta función nos devuelve un
código que guardamos en un int)
  if(codigo_respuesta>0){
    //Serial.println("Código HTTP ► " );   //Print return code
    //Serial.println(codigo_respuesta);
    if(codigo_respuesta == 200){
      String cuerpo_respuesta = http.getString();
      Serial.println("El servidor respondió ▼ ");
      Serial.println(cuerpo_respuesta);
      }
  }else{
    Serial.print("Error enviando POST, código: ");
    Serial.println(codigo_respuesta);
  }
  http.end();  //libero recursos
}

void uploadPump(int tankState, int pumpPreparation, int pumpState, int
heaterState){
  HTTPClient http;
  WiFiClient client;
  String tS = String(tankState);
  String pP = String(pumpPreparation);
  String pS = String(pumpState);
  String datos_a_enviar = "tankState=" + tS + "&pumpPrep=" + pP +
"&pumpState=" + pS;
  http.begin(client,                 "http://172.16.2.131/postPump.php");
//Indicamos el destino
  http.addHeader("Content-Type",    "application/x-www-form-urlencoded");
//Preparamos el header text/plain si solo vamos a enviar texto plano sin
un paradigma llave:valor.
  float codigo_respuesta = http.POST(datos_a_enviar);   //Enviamos el post
pasándole, los datos que queremos enviar. (esta función nos devuelve un
código que guardamos en un int)
  if(codigo_respuesta>0){
    //Serial.println("Código HTTP ► " );   //Print return code
    //Serial.println(codigo_respuesta);
    if(codigo_respuesta == 200){
```

```
      String cuerpo_respuesta = http.getString();
      Serial.print("El servidor respondió: ");
      Serial.println(cuerpo_respuesta);
      }
  }else{
    Serial.print("Error enviando POST, código: ");
    Serial.println(codigo_respuesta);
  }
  http.end();  //libero recursos
}
int esp_recievePumpTime(void){
  //Formato que se recibe la hora XXYYZZ
  //XX->Minutos de inicio; YY->Duracion; ZZ->Hora inicio
  WiFiClient client;
  long valor;
  if (client.connect(host, 80)) {
    //Serial.println("Conectado al servidor en la Raspberry Pi");
    // Enviar solicitud GET al servidor en la Raspberry Pi
    client.println("GET /getHourPump.php HTTP/1.1");
    client.println("Host: " + String(host));
    client.println("Connection: close");
    client.println();
    // Leer y guardar la respuesta del servidor en la Raspberry Pi
    String response = "";
    while (client.connected() || client.available()) {
      if (client.available()) {
        char c = client.read();
        response += c;  // Concatenar el caracter a la respuesta
      }
    }
    client.stop();
    //Serial.println("\nDesconectado del servidor en la Raspberry Pi");
    // Buscar el índice donde comienza el valor numérico en la respuesta
    int startIndex = response.lastIndexOf("\n") + 1;
    // Extraer la línea que contiene el valor numérico
    String numericLine = response.substring(startIndex);
    // Convertir la línea a un valor numérico
    valor = numericLine.toInt();
    // Imprimir el valor numérico en el monitor serial
    /*Serial.print("Valor recibido: ");
    Serial.println(valor);*/
  } else {
    //Serial.println("Error al conectar al servidor en la Raspberry Pi");
  }
  return valor;
}
int esp_recieveHeatTime(void){
  //Formato que se recibe la hora XXYYZZ
  //XX->Minutos de inicio; YY->Duracion; ZZ->Hora inicio
  WiFiClient client;
  long valor;
  if (client.connect(host, 80)) {
    //Serial.println("Conectado al servidor en la Raspberry Pi");
    // Enviar solicitud GET al servidor en la Raspberry Pi
    client.println("GET /getHourHeat.php HTTP/1.1");
    client.println("Host: " + String(host));
```

```
    client.println("Connection: close");
    client.println();
    // Leer y guardar la respuesta del servidor en la Raspberry Pi
    String response = "";
    while (client.connected() || client.available()) {
      if (client.available()) {
        char c = client.read();
        response += c;  // Concatenar el caracter a la respuesta
      }
    }
    client.stop();
    //Serial.println("\nDesconectado del servidor en la Raspberry Pi");
    // Buscar el índice donde comienza el valor numérico en la respuesta
    int startIndex = response.lastIndexOf("\n") + 1;
    // Extraer la línea que contiene el valor numérico
    String numericLine = response.substring(startIndex);
    // Convertir la línea a un valor numérico
    valor = numericLine.toInt();
    // Imprimir el valor numérico en el monitor serial
    /*Serial.print("Valor recibido: ");
    Serial.println(valor);*/
  } else {
    //Serial.println("Error al conectar al servidor en la Raspberry Pi");
  }
  return valor;
}
```

Anexo 3. Código de la página principal

```
<?php

DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se excribe
localhost
DEFINE ('DB_NAME', 'Invernadero');

$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);

/*if($db_con){
    echo "Connection OK";
}else{
    echo "Connection failed".mysqli_connect_error();
}*/
?>
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="styles.css">
    <title>Invernadero TFG</title>
</head>
<body>
    <h1>INVERNADERO TFG</h1>
    <table id="table1">
        <tr id="encabezado">
            <td>Sensor</td>
            <td id="valor1">Valor</td>
            <td>Unidades</td>
        </tr>
        <!--Fila temperatura-->
        <tr>
            <?php
                $sqlTemp = "SELECT * FROM Temperatura ORDER BY ID DESC
LIMIT 1;";
                $resultTemp = mysqli_query($db_con, $sqlTemp);
                while($mostrarTemp = mysqli_fetch_array($resultTemp)){
            ?>
            <td>Temperatura</td>
            <td id="valor1"><?php echo $mostrarTemp['Temp'] ?></td>
            <td>ºC</td>
            <?php
                }
            ?>
        </tr>
        <!--Fila Humedad-->
        <tr>
            <?php
                $sqlMois = "SELECT * FROM Humedad ORDER BY ID DESC LIMIT
1;";
```

```
                $resultMois = mysqli_query($db_con, $sqlMois);
                while($mostrarMois = mysqli_fetch_array($resultMois)){
            ?>
            <td>Humedad</td>
            <td id="valor1"><?php echo $mostrarMois['Mois'] ?></td>
            <td>%</td>
            <?php
                }
            ?>
        </tr>
        <!--Fila Nivel-->
        <tr>
            <?php
                $sqlLevel = "SELECT * FROM Nivel ORDER BY ID DESC LIMIT
1;";
                $resultLevel = mysqli_query($db_con, $sqlLevel);
                while($mostrarLevel = mysqli_fetch_array($resultLevel)){
            ?>
            <td>Nivel</td>
            <td id="valor1"><?php echo $mostrarLevel['Level'] ?></td>
            <td>cm</td>
            <?php
                }
            ?>
        </tr>
    </table>
    <!--Tabla del estado del riego-->
    <table id="table2">
        <tr id="encabezado">
           <td>Estado Tanque</td>
           <td>Bomba preparada</td>
           <td>Estado Bomba</td>
        </tr>
        <tr id="estado2">
            <?php
                $sqlPump = "SELECT * FROM Bomba ORDER BY ID DESC LIMIT 1;";
                $resultPump = mysqli_query($db_con, $sqlPump);
                while($mostrarPump = mysqli_fetch_array($resultPump)){
            ?>
            <td><?php echo $mostrarPump['Tank'] ?></td>
            <td><?php echo $mostrarPump['Prep'] ?></td>
            <td><?php echo $mostrarPump['Pump'] ?></td>
            <?php
                }
            ?>
        </tr>
    </table>
    <div>
        <input  id="refresh"  type="button"  value="Actualizar  valores"
onclick="javascript:window.location.reload();"/>
    </div>
    <div id="formulario"><table>
        <h4>Programar horas de riego</h4>
        <form action="postTime.php" method="post">
            <label for="Pump_Init_Time">Hora de inicio: </label>
```

```
            <input  type="number"  id="pumpInitHour"  name="pumpInitHour"
min="1" max="24">
            <label>:</label>
            <input   type="number"   id="pumpInitMin"   name="pumpInitMin"
min="00" max="59">
            <label for="Pump_Duration">Duración: </label>
            <select name="Pump_Duration" id="Pump_Duration">
                <option value="10">00:10h</option>
                <option value="15">00:15h</option>
                <option value="20">00:20h</option>
                <option value="30">00:30h</option>
                <option value="45">00:45h</option>
                <option value="60">01:00h</option>
                <option value="75">01:15h</option>
                <option value="90">01:30h</option>
            </select>
            <br>
            <label for="Heat_Init_Time">Hora de inicio: </label>
            <input  type="number"  id="heatInitHour"  name="heatInitHour"
min="1" max="24">
            <label>:</label>
            <input   type="number"   id="heatInitMin"   name="heatInitMin"
min="00" max="59">
            <label for="Heat_Duration">Duración: </label>
            <select name="Heat_Duration" id="Heat_Duration">
                <option value="10">00:10h</option>
                <option value="15">00:15h</option>
                <option value="20">00:20h</option>
                <option value="30">00:30h</option>
                <option value="45">00:45h</option>
                <option value="60">01:00h</option>
                <option value="75">01:15h</option>
                <option value="90">01:30h</option>
            </select>
            <br>
            <button>Confirmar</button>
        </form>
    </div>
    <div id="horario">
        <?php
          $sqlPumpTime = "SELECT * FROM Horarios ORDER BY ID DESC LIMIT
1;";
          $resultPumpTime = mysqli_query($db_con, $sqlPumpTime);
          while($mostrarPumpTime = mysqli_fetch_array($resultPumpTime)){
        ?>
        <p>Inicio del riego: <?php echo $mostrarPumpTime['BombaHour'] ?> :
<?php echo $mostrarPumpTime['BombaMin'] ?></p>
        <p>Duracion         del         riego:         <?php         echo
$mostrarPumpTime['BombaDuration'] ?></p>
        <p>Inicio      de      la      calefacción:      <?php      echo
$mostrarPumpTime['CalefacHour']       ?>       :       <?php       echo
$mostrarPumpTime['CalefacMin'] ?></p>
        <p>Duracion     de     la     calefacción:     <?php     echo
$mostrarPumpTime['CalefacDuration'] ?> <?php } ?></p>
    </div>
</body>
```

```
</html>
```

Anexo 4. Códigos PHP

Código 1. Archivo “getHourHeat.php”

Este código se utiliza para obtener la hora programada para encender el calefactor almacenada en la base de datos.

```
<?php
DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se escribe
localhost
DEFINE ('DB_NAME', 'Invernadero');
$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);//Conexion
con la bbdd
//Se comprueba que la conexión se ha establecido
/*if($db_con){
    echo "Connection OK";
}else{
    echo "Connection failed".mysqli_connect_error();
}*/

$sql = "SELECT * FROM Horarios ORDER BY ID DESC LIMIT 1;";//Sentencia de
solicitud
$result = mysqli_query($db_con, $sql);

// Configurar encabezados para permitir acceso desde cualquier origen
(CORS)
header("Access-Control-Allow-Origin: *");
header("Content-Type: text/plain");

// Enviar los datos como respuesta al cliente (ESP8266)
while($mostrar = mysqli_fetch_array($result)){
    echo $mostrar['CalefacHour'];
    echo $mostrar['CalefacMin'];
    echo $mostrar['CalefacDuration'];
}
?>
```

Código 2. Archivo “getHourPump.php”

Código utilizado para obtener la hora programada para encender la bomba almacenada en la base de datos.

```
<?php
DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se escribe
localhost
DEFINE ('DB_NAME', 'Invernadero');
$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);//Conexion
con la bbdd
//Se comprueba que la conexión se ha establecido
```

```
/*if($db_con){
    echo "Connection OK";
}else{
    echo "Connection failed".mysqli_connect_error();
}*/

$sql = "SELECT * FROM Horarios ORDER BY ID DESC LIMIT 1;";//Sentencia de
solicitud
$result = mysqli_query($db_con, $sql);

// Configurar encabezados para permitir acceso desde cualquier origen
(CORS)
header("Access-Control-Allow-Origin: *");
header("Content-Type: text/plain");

// Enviar los datos como respuesta al cliente (ESP8266)
while($mostrar = mysqli_fetch_array($result)){
    echo $mostrar['BombaHour'];
    echo $mostrar['BombaMin'];
    echo $mostrar['BombaDuration'];
}
?>
```

Código 3. Archivo “postLevel.php”

Código utilizado para almacenar los valores de nivel obtenidos por el sensor en la base de datos.

```
<?php
$level = $_POST['level'];//Obtener el valor por el metodo POST
DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se escribe
localhost
DEFINE ('DB_NAME', 'Invernadero');
$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);//Conexion
con la bbdd
//Se comprueba que la conexión se ha establecido
/*if($db_con){
    echo "Connection OK";
}else{
    echo "Connection failed".mysqli_connect_error();
}*/
$sql = "INSERT INTO Nivel (ID, Level) VALUES (NULL,
".$level.")";//Sentencia de solicitud
//Mostar respuesta
if(mysqli_query($db_con,$sql)===TRUE){
    echo "New record created succesfully";
}else{
    echo "ERROR: ".mysqli_error($db_con);
}
mysqli_close($db_con);
?>
```

Código 4. Archivo “postLight.php”

Código utilizado para almacenar los valores de luz obtenidos por el sensor en la base de datos.

```
<?php
$light = $_POST['light'];//Obtener el valor por el metodo POST
DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se excribe
localhost
DEFINE ('DB_NAME', 'Invernadero');
$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);//Conexion
con la bbdd
//Se comprueba que la conexión se ha establecido
/*if($db_con){
    echo "Connection OK";
}else{
    echo "Connection failed".mysqli_connect_error();
}*/
$sql = "INSERT INTO Luz (Luminosidad, Tiempo) VALUES (NULL,
".$light.")";//Sentencia de solicitud
if(mysqli_query($db_con,$sql)===TRUE){
    echo "New record created succesfully";
}else{
    echo "ERROR: ".mysqli_error($db_con);
}
mysqli_close($db_con);
?>
```

Código 5. Archivo “postMois.php”

Código utilizado para almacenar los valores de humedad obtenidos por el sensor en la base de datos.

```
<?php
$moisture = $_POST['mois'];//Obtener el valor por el metodo POST
DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se escribe
localhost
DEFINE ('DB_NAME', 'Invernadero');
$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);//Conexion
con la bbdd
//Se comprueba que la conexión se ha establecido
/*if($db_con){
    echo "Connection OK";
}else{
    echo "Connection failed".mysqli_connect_error();
}*/
$sql = "INSERT INTO Humedad (ID, Mois) VALUES (NULL,
".$moisture.")";//Sentencia de solicitud
if(mysqli_query($db_con,$sql)===TRUE){
    echo "New record created succesfully";
}else{
```

```
    echo "ERROR: ".mysqli_error($db_con);
}
mysqli_close($db_con);
?>
```

Código 6. Archivo “postPump.php”

Código utilizado para almacenar el estado de la bomba en la base de datos.

```
<?php
//Obtener los valores por el metodo POST
$tState = $_POST['tankState'];
$pReady = $_POST['pumpPrep'];
$pAct = $_POST['pumpState'];
DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se escribe
localhost
DEFINE ('DB_NAME', 'Invernadero');
$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);//Conexion
con la bbdd
//Se comprueba que la conexión se ha establecido
/*if($db_con){
    echo "Connection OK";
}else{
    echo "Connection failed".mysqli_connect_error();
}*/
$sql = "INSERT INTO Bomba (ID, Tank, Prep, Pump) VALUES (NULL,
".$tState.",".$pReady.",".$pAct.")";//Sentencia de solicitud
//Mostar respuesta
if(mysqli_query($db_con,$sql)===TRUE){
    echo "New record created succesfully";
}else{
    echo "ERROR: ".mysqli_error($db_con);
}
mysqli_close($db_con);
?>
```

Código 6. Archivo “postTemp.php”

Código utilizado para almacenar los valores de temperatura obtenidos por el sensor en la base de datos.

```
<?php
$temp = $_POST['temp'];//Obtener el valor por el metodo POST
DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se excribe
localhost
DEFINE ('DB_NAME', 'Invernadero');
$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);//Conexion
con la bbdd
//Se comprueba que la conexión se ha establecido
/*if($db_con){
    echo "Connection OK";
```

```
}else{
    echo "Connection failed".mysqli_connect_error();
}*/
$sql = "INSERT INTO Temperatura (ID, Temp) VALUES (NULL,
".$temp.")";//Sentencia de solicitud
//Mostar respuesta
if(mysqli_query($db_con,$sql)===TRUE){
    echo "New record created succesfully";
}else{
    echo "ERROR: ".mysqli_error($db_con);
}
mysqli_close($db_con);
?>
```

Código 7. Archivo “postTime.php”

Código utilizado para almacenar la programación del tiempo para el riego automático y el encendido de la calefacción en la base de datos.

```
<?php
//Obtener el valor por el metodo POST
$initHourP = $_POST['pumpInitHour'];
$initMinP = $_POST['pumpInitMin'];
$durationP = $_POST['Pump_Duration'];
$initHourH = $_POST['heatInitHour'];
$initMinH = $_POST['heatInitMin'];
$durationH = $_POST['Heat_Duration'];
DEFINE ('DB_USER', 'root');
DEFINE ('DB_PSWD', 'late');
DEFINE ('DB_HOST', 'localhost'); //No se escribe la IP, se excribe
localhost
DEFINE ('DB_NAME', 'Invernadero');
$db_con = mysqli_connect(DB_HOST, DB_USER, DB_PSWD, DB_NAME);//Conexion
con la bbdd
/*if($db_con){
    echo "Connection OK";
}else{
    echo "Connection failed".mysqli_connect_error();
}*/
$sql = "INSERT INTO Horarios (ID, BombaHour, BombaMin, BombaDuration,
CalefacHour, CalefacMin, CalefacDuration) VALUES
(NULL,".$initHourP.",".$initMinP.",".$durationP.",".$initHourH.",".$initM
inH.",".$durationH.")";//Sentencia de solicitud
//Mostrar respuesta
if(mysqli_query($db_con,$sql)===TRUE){
    echo "New record created succesfully";
}else{
    echo "ERROR: ".mysqli_error($db_con);
}
mysqli_close($db_con);
?>
<br>
<!--Boton para volver a la página inicial-->
<button onclick="location.href='index.php'">Volver</button>
```

UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

Grado en Ingeniería Electrónica Industrial y Automática

Curso 2022/2023

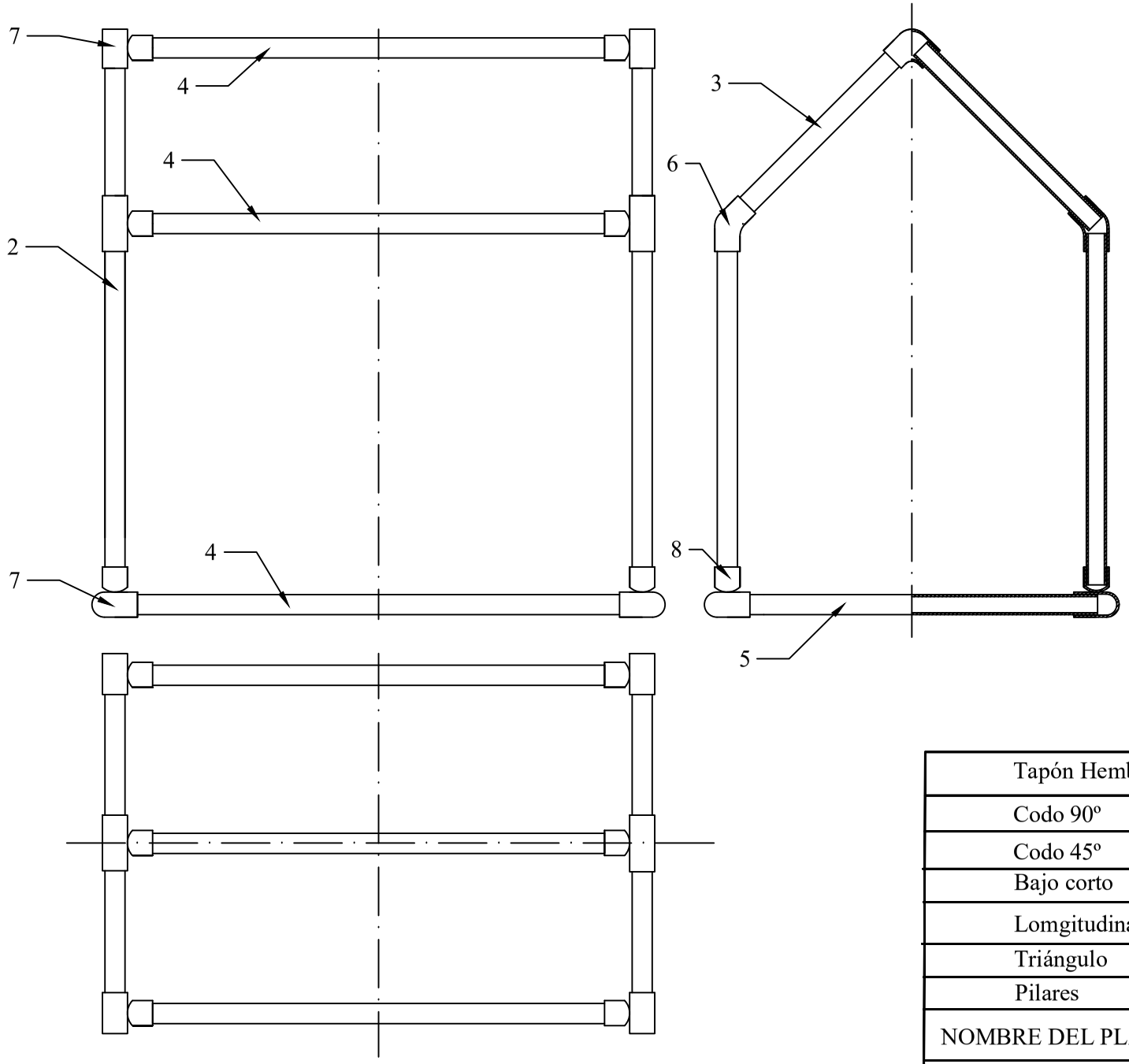
TRABAJO DE FIN DE GRADO:

Diseño e implementación de un sistema de control electrónico de germinación de semillas en un miniinvernadero.

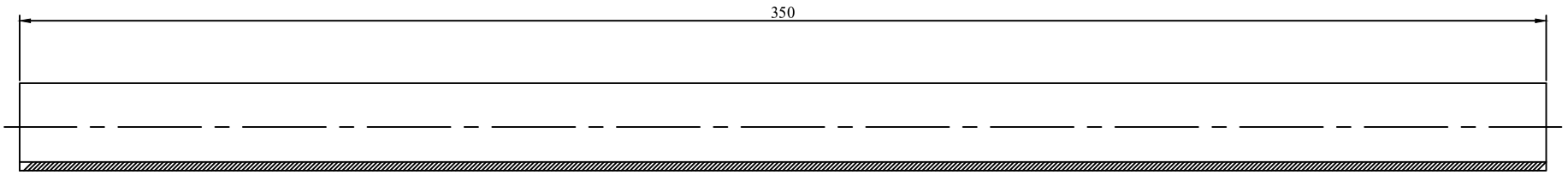
DOCUMENTO 2. PLANOS

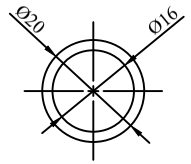
AUTOR: Adrián Guillem Vallés
TUTOR: Dr. Enrique Berjano Zanón

Índice de contenido

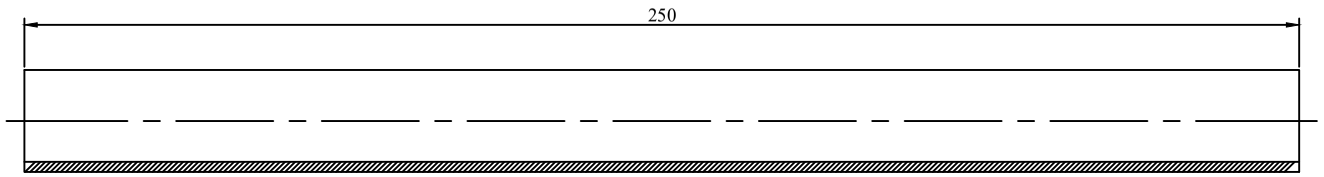


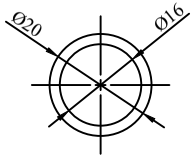
Tapón Hembra	10	8
Codo 90º	6	7
Codo 45º	4	6
Bajo corto	2	5
Lomgitudinal	5	4
Triángulo	4	3
Pilares	4	2
NOMBRE DEL PLANO	UNIDADES	NÚMERO
Autor: Adrián Guillem Vallés	Denominación del plano: Estructura	Escala: 1:4 / Hoja: 1



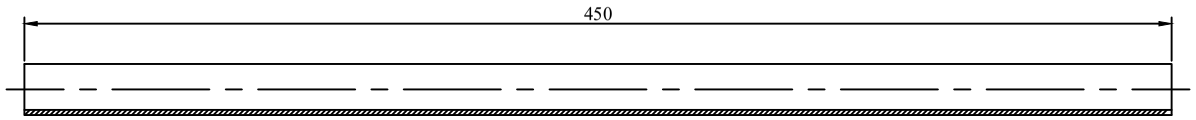


AUTOR	PLANO	ESCALA	HOJA
Adrián Guillem Vallés	Pilar	1:1	2

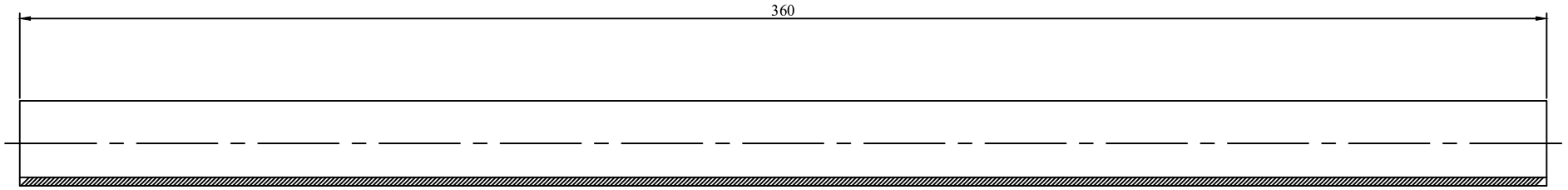


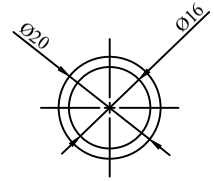


AUTOR	PLANO	ESCALA	HOJA
Adrián Guillem Vallés	Triángulo	1:1	3

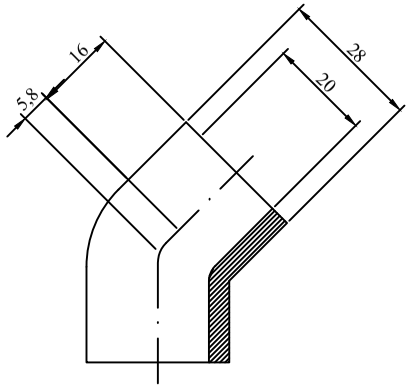


AUTOR	PLANO	ESCALA	HOJA
Adrián Guillem Vallés	Longitudinal	1:2	4

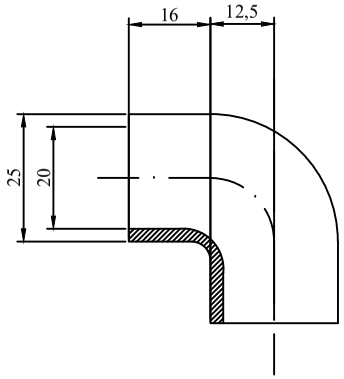




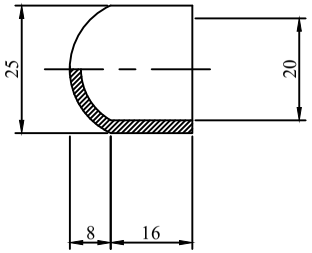
AUTOR	PLANO	ESCALA	HOJA
Adrián Guillem Vallés	Corto Bajo	1:1	5



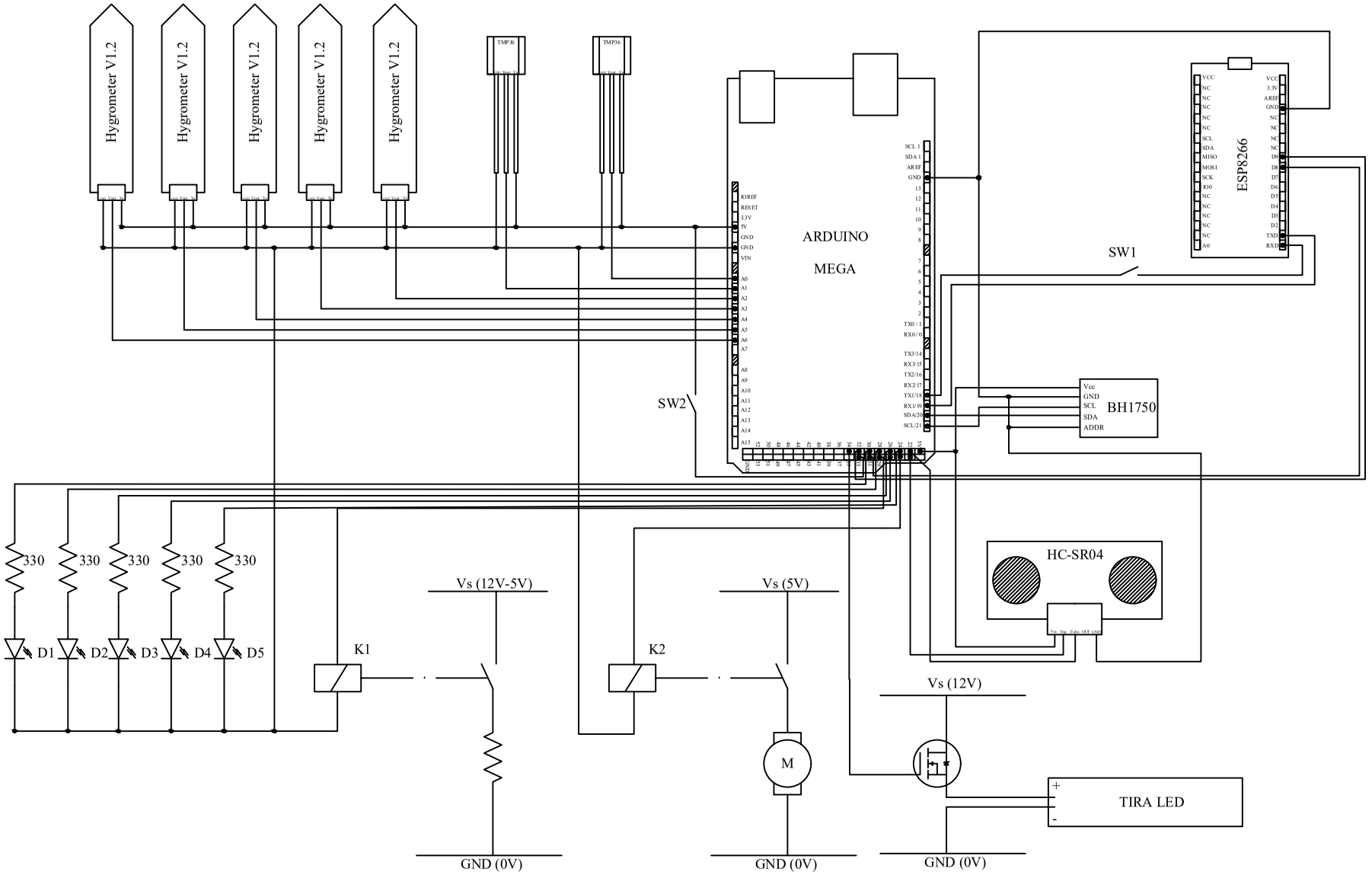
AUTOR	PLANO	ESCALA	HOJA
Adrián Guillem Vallés	Codo 45º	1:1	6



AUTOR	PLANO	ESCALA	HOJA
Adrián Guillem Vallés	Codo 90º	1:1	7



AUTOR	PLANO	ESCALA	HOJA
Adrián Guillem Vallés	Tapón Hembra	1:1	8



AUTOR	PLANO	HOJA
Adrián Guillem Vallés	Esquema eléctrico	9

UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

Grado en Ingeniería Electrónica Industrial y Automática

Curso 2022/2023

TRABAJO DE FIN DE GRADO:

Diseño e implementación de un sistema de control electrónico de germinación de semillas en un mini invernadero.

DOCUMENTO 3. PLIEGO DE CONDICIONES

AUTOR: Adrián Guillem Vallés
TUTOR: Dr. Enrique Berjano Zanón

Índice de contenido

1. Alcance

Este pliego de condiciones se refiere al sistema diseñado previamente. Se han analizado los materiales que se van a utilizar. Se ha definido un modelo de montaje e instalación de todo el sistema. Sin embargo, la maquinaria no ha sido incluida en el documento. En caso de conflicto con lo mencionado anteriormente, prevalecerá lo mencionado en los documentos anteriores.

2. Materiales

2.1. Plástico PVC

Se utilizará plástico PVC para la construcción de los cimientos de la estructura. El diámetro de los tubos y sus respectivas medidas vienen especificados en el documento *2. Planos.* La normativa que rige la calidad y la composición del material es la UNE-EN-ISO 1452 PN 16 y ha de ser cumplida por el fabricante asegurando las correctas propiedades del material utilizado.

2.2. Plástico para invernadero

Para cubrir la zona de cultivo se va a utilizar plástico especial para invernadero. El fabricante deberá asegurar que el plástico se ha fabricado con todas las propiedades necesarias para su uso correcto en estos proyectos.

2.3. Componentes

Los valores de las resistencias han de ser comprobados mediante un multímetro digital. Las resistencias tienen un valor de tolerancia de un 5%. Se debe realizar una prueba del valor de la resistencia para comprobar que dicha tolerancia se cumple en cada una de las resistencias.

Las placas Arduino Mega y ESP8266 usadas en el proyecto presentan características que debe certificar el fabricante, asegurando el correcto funcionamiento y los niveles de seguridad indicados en la normativa correspondiente.

2.4. Control de calidad

El fabricante de cada uno de los materiales y componentes debe especificar que las características y propiedades han sido cumplidas según la normativa que corresponda en el certificado de calidad.

3. Instalación

3.1. Montaje y conexiones

El proceso de instalación y montaje de todos los elementos es un proceso que se ha de cumplir conforme a lo especificado en el documento 1. Memoria, en el apartado 6.1. Estructura. Del mismo modo, las conexiones de todo el sistema electrónico han de ser conectadas según el esquema eléctrico que se especifica en el documento 2. Planos.

La instalación y programación del software encargado del control web y de los microcontroladores no será realizado por el usuario ya que vendrán dispuestos de serie según las necesidades que haya proporcionado el usuario previamente.

3.2. Control de calidad

Previamente a la entrega del producto final, se realizará una prueba de funcionamiento para comprobar que todas las conexiones se han realizado de forma correcta y su funcionamiento es el esperado.

UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

Grado en Ingeniería Electrónica Industrial y Automática

Curso 2022/2023

TRABAJO DE FIN DE GRADO:

Diseño e implementación de un sistema de control electrónico de germinación de semillas en un miniinvernadero.

DOCUMENTO 4. PRESUPUESTO

AUTOR: Adrián Guillem Vallés
TUTOR: Dr. Enrique Berjano Zanón

Índice de contenido

1. Alcance

El presupuesto es una forma de cuantificar correctamente y con todo detalle el coste del proyecto. En él se especifican el coste de cada uno de los componentes que se utilizan para diseñar el sistema, así como el gasto en conjunto de la maqueta, el coste de la mano de obra en el proceso de diseño y los costes extras que se puedan añadir como pueden ser: la amortización de equipos (portátiles, impresoras, etc.), costes en luz, agua e internet, gastos en material de oficina, etc.

2. Precios unitarios

Primeramente, se va a mostrar el primer apartado del presupuesto en el que se especifican los precios por unidad de cada uno de los elementos utilizados en el diseño del sistema de control.

1. Precios unitarios			
Ref	Unidad	Componente	Precio(€)
Materiales			
m1	u.	Arduino Mega	41,55 €
m2	u.	ESP8266	6,90 €
m3	u.	TMP36	1,91 €
m4	u.	Semilleros	6,67 €
m5	u.	FC28	2,80 €
m6	u.	BH1750	2,40 €
m7	u.	Sensor ultrasonidos	2,90 €
m8	u.	Bomba de agua	14,99 €
m9	u.	Calefactor	13,36 €
m10	m.	Tira de leds	3,99 €
m11	u.	Reles	2,03 €
m12	u.	Transistores IRF3205LPBF	1,49 €
m13	u.	Resistencias	0,05 €
m14	m.	Tubos PVC 20 mm Rectos	2,09 €
m15	u.	Codos 90º PVC - 20 mm diametro	0,59 €
m16	u.	Codos 45º PVC - 20 mm diametro	0,45 €
m17	u.	Tapones hembra PVC - 20 mm diametro	0,79 €
m18	u.	Cables	0,05 €
m19	m.	Plastico para invernadero	1,20 €
m20	u.	Raspberry Pi	60,00 €
m21	u.	Regletas	0,05 €
m22	u.	Cable de alimentación USB	5,00 €
m23	m.	Goma de riego	0,26 €

m24	u.	Goteo de agua 2L/H	0,35 €
m25	u.	Caja eléctrica	4,86 €
Mano de obra			
h1	h.	Ingeniero técnico	15,00 €
Equipos			
e1	u.	Portatil	260,00 €
Otros costes			
c1	€/año	Licencia AutoCAD	140,00 €
c2	€/kWh	Electricidad	0,17 €
c3	€/mes	Wi-Fi	15,00 €
c4	€/mes	Material de oficina	10,00 €

3. Precios unitarios por desglose

En este apartado se separan en cuatro grupos según correspondan a cada una de las partes que tiene este proyecto, es decir, se dividen la maqueta, la mano de obra, los equipos y otros costes; y, a su vez, la maqueta se divide en tres subgrupos: estructura, circuito electrónico y sistema de riego.

Precios unitarios por desglose					
Ref.	Unidad	Descripción	Precio	Cantidad	Total
d1	u.	Estructura			
Materiales					
m4	u.	Semilleros	6,67 €	2	13,34 €
m14	m.	Tubos PVC 20 mm Rectos	2,09 €	6	12,54 €
m15	u.	Codos 90º PVC - 20 mm diámetro	0,59 €	6	3,54 €
m16	u.	Codos 45º PVC - 20 mm diámetro	0,45 €	4	1,80 €
m17	u.	Tapones hembra PVC - 20 mm diámetro	0,79 €	10	7,90 €
m20	m.	Plástico para invernadero	1,20 €	5	6,00 €
			TOTAL		45,12 €
d2	u.	Circuito electrónico			
Materiales					
m1	u.	Arduino Mega	41,55 €	1	41,55 €
m2	u.	ESP8266	6,90 €	1	6,90 €
m3	u.	TMP36	1,91 €	2	3,82 €
m5	u.	FC28	2,80 €	5	14,00 €
m6	u.	BH1750	2,40 €	1	2,40 €
m7	u.	Sensor ultrasonidos	2,90 €	1	2,90 €
m9	u.	Calefactor	13,36 €	1	13,36 €
m10	m.	Tira de leds	3,99 €	0,4	1,60 €

m11	u.	Relés	2,03 €	2	4,06 €
m12	u.	Transistores IRF3205LPBF	1,49 €	1	1,49 €
m13	u.	Resistencias	0,05 €	3	0,15 €
m18	u.	Cables	0,05 €	200	10,49 €
m20	u.	Raspberry Pi	60,00 €	1	60,00 €
m21	u.	Regletas	0,05 €	4	0,20 €
m22	u.	Cable de alimentación USB	5,00 €	2	10,00 €
m23	m.	Goma de riego	0,26 €	1,5	0,39 €
m24	u.	Goteo de agua 2L/H	0,35 €	36	12,42 €
m25	u.	Caja eléctrica	4,86 €	1	4,86 €
			TOTAL		190,59 €

d3	u.	Sistema de riego			
Materiales					
m8	u.	Bomba de agua	14,99 €	1	14,99 €
m23	m.	Goma de riego	0,26 €	2	0,52 €
m24	u.	Goteo de agua 2L/H	0,35 €	36	12,42 €
			TOTAL		27,93 €

p1	u.	Maqueta			
d1	u.	Estructura	45,12 €	1	45,12 €
d2	u.	Circuito electrónico	190,59 €	1	190,59 €
d3	u.	Sistema de riego	27,93 €	1	27,93 €
Impuestos					
	%	Precio sin IVA	-	-	263,64 €
	%	Precio con IVA	21,00%	55,36 €	319,00 €
			Coste de la maqueta		319,00 €

p2	u.	Mano de obra			
h1	h.	Ingeniero técnico	15,00 €	300	4.500,00 €
Impuestos					
	%	Precio sin IVA	-	-	4.500,00 €
	%	Precio con IVA	21,00%	945,00 €	5.445,00 €
			Coste de la mano de obra		5.445,00 €

Ref	Unidad	Descripcion	Precio	Amortización	Precio por año	Precio por maqueta
p3	u.	Equipos				
e1	u.	Portátil	260,00 €	10	26,00 €	26,00 €
Impuestos						
	%	Precio sin IVA		-	-	26,00 €

	%	Precio con IVA	21,00%	5,46 €	31,46 €
			Coste de equipos		31,46 €

p4	u.	Otros costes				
c1	€/año	Licencia AutoCAD	140,00 €	0,15625	21,88 €	21,88 €
c2	€/kWh	Electricidad	0,17 €	300	51,00 €	51,00 €
c3	€/mes	Wifi	15,00 €	1,875	28,13 €	28,13 €
c4	€/mes	Material de oficina	10,00 €	-	10,00 €	10,00 €
Impuestos						
	%	Precio sin IVA		-	-	111,00 €
	%	Precio con IVA		21,00%	23,31 €	134,31 €
				Otros costes		134,31 €

4. Medidas

Seguidamente se agrupan los cuatro grupos nombrados anteriormente y se especifican la cantidad de cada uno. En este caso, como solo se ha montado una maqueta, se utilizará una única unidad de cada grupo.

Medidas			
Ref	**Unidad**	**Descripción**	**Cantidad**
p1	u.	Maqueta	1
p2	u.	Mano de obra	1
p3	u.	Equipos	1
p4	u.	Otros costes	1

5. Valuación

Finalmente, se juntan las cantidades que se van a presupuestar junto con el coste que supone cada uno de los grupos y se obtiene el valor final del presupuesto.

Valuación					
Ref	**Unidad**	**Descripción**	**Precio (€)**	**Cantidad**	**Total**
p1	u.	Maqueta	319,00 €	1	319,00 €
p2	u.	Mano de obra	5.445,00 €	1	5.445,00 €
p3	u.	Equipos	31,46 €	1	31,46 €
p4	u.	Otros costes	134,31 €	1	134,31 €
			Total presupuesto		5.929,77 €

6. Conclusión

El coste total presupuestado puede resultar un tanto elevado debido a la cantidad de investigación que requiere esta primera maqueta elevando así el coste de la mano de obra. El producto final tendría un precio de cara al público entre 350€ y 400€ en el caso de que fuese exitoso y saliese al mercado. Este precio sería más asequible para un usuario agricultor que no se dedique profesionalmente a esta profesión ya que con los años el valor del miniinvernadero será amortizado.