UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Control electrónico de un invernadero

Trabajo Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

AUTOR/A: Cedeño Vinces, Adair Enrique

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:

CONTROL ELECTRÓNICO DE UN INVERNADERO

DOCUMENTO 1. MEMORIA

AUTOR: Adair Enrique Cedeño Vinces

TUTOR: Dr. Enrique Berjano Zanón

Índice

Documento 1. Memoria

Documento 2. Planos

Documento 3. Pliego de condiciones

Documento 4. Presupuesto

RESUMEN

Debido al cambio climático las sequías e fuertes cambios de temporal están afectando a la producción agrícola de todo el mundo. Para paliar los efectos negativos de estos agentes naturales es más necesario que nunca el desarrollo de tecnologías de invernaderos, instalaciones dedicadas a la plantación controlada en un ambiente controlado y óptimo para el desarrollo de las plantas.

El presente proyecto estudia las necesidades de diseño de un invernadero, tanto en las bases de su construcción como en la tecnología de ámbito electrónico que presenta para un control automatizado. Además, presenta también un modelo a escala, una maqueta funcional con el mismo tipo de tecnología, así como el desarrollo de una interfaz gráfica para el control del sistema realizado.

1. Objetivo

El objetivo del presente documento es plantear y desarrollar el diseño de un invernadero automatizado funcional.

Antes de comenzar a trabajar con el diseño, se debe entender el funcionamiento de un invernadero, sus características y las condiciones en las que permite un óptimo crecimiento de los cultivos.

Para el estudio de la funcionalidad de este se va a proponer el diseño de una maqueta a escala, intentando mantener su funcionalidad, para su posterior construcción y puesta a prueba.

Se debe trabajar en dos áreas de desarrollo tanto en el diseño inicial como el diseño a escala; el sistema de control y monitorización y el diseño estructural del invernadero.

Para el sistema de control se estudiarán las diferentes opciones de controladores para utilizar como plataforma, teniendo en cuenta las limitaciones de hardware y software. Con la plataforma de control elegida se desarrollará un código modular para la utilización del sistema de control en diferentes casos de la misma índole, con variaciones en cuanto a la monitorización y control disponible de cada programa. Esto está relacionado con la otra área de desarrollo, debido a que un diferente diseño estructural puede permitir la ampliación tanto del número de sensores y actuadores como de magnitudes y características a monitorizar y controlar.

Con el objetivo de la manipulación y la observación de los diferentes parámetros del programa, se debe proponer una interfaz de control, por lo que se estudiarán las diferentes opciones para este propósito y se trabajará con la más adecuada para este caso.

Por último, se va a realizar el montaje de la solución adoptada para la puesta en marcha del sistema y la comprobación del correcto funcionamiento del diseño a escala. Se plantearás los errores del sistema y los márgenes en los que puede trabajar, teniendo en cuenta las limitaciones del montaje y los componentes utilizados.

2. Estudio de necesidades

Durante la historia, el ser humano ha necesitado de una producción agrícola para subsistir, peleando contra los efectos naturales del clima. Este motivo impulsa el creamiento de una tecnología, los invernaderos como sistema de cultivos protegidos.

Los invernaderos son estructuras artificiales creadas con el propósito de aislar y proteger una plantación de los efectos meteorológicos. Además, permiten el control de algunos de los factores que dan paso al desarrollo óptimo de estos cultivos.

Con el actual panorama climatológico, proveniente del cambio climático, se han visto dañados innumerables cultivos. Este factor puede agravar la situación que se vive en el mundo de hambruna, debido a la disminución de las cosechas. La creación de invernaderos protege los cultivos y ayuda a un correcto desarrollo de las plantas, paliando así los efectos negativos del cambio climatológico. El desarrollo de esta tecnología puede ayudar a lograr el objetivo de hambre cero, uno de los objetivos de desarrollo sostenible. Con la creación de invernaderos se puede crear plantaciones en terrenos más áridos, donde es más complicada la producción agrícola generando así comida para acabar con el hambre en el mundo.

Los sistemas de control se basan principalmente en cuatro parámetros, la humedad, la luminosidad, la temperatura y la calidad del aire. Adicionalmente se puede controlar de igual manera la calidad y los nutrientes del terreno, sin embargo, no es una tarea para la que está concebida inicialmente la estructura de un invernadero.

Por otro lado, el armazón de la instalación consta de pilares y vigas sobre el que se añade la cubierta. Esta estructura exterior busca rigidez, aguantando los elementos climatológicos y creando unas condiciones de temperatura y humedad independientes a las exteriores.

La tecnología de los invernaderos se estudia ampliamente para analizar como los criterios de diseño de la estructura, los materiales y las dimensiones influyen enormemente en la optimización de los factores de crecimiento de las plantas [1]. Además de ser una instalación que pueda variar mucho dependiendo del tipo de vegetal plantado.

Para que un invernadero sea funcional debe cumplir al menos con las siguientes características:

- La estructura exterior debe tener una cubierta que permita el paso de los rayos del sol, siendo de un material transparente. Además, debe poder aislar la temperatura y la humedad en su interior. Esta condición aprovecha que las plantas y el suelo reciben la radiación solar, calentando el interior y permitiendo un correcto proceso de fotosíntesis. Sin embargo, la fuente calor y de iluminación puede ser artificial en caso de ser necesitado.
- En caso de haber un exceso de temperatura, un invernadero cuenta con ventilación. De esta manera se logra enfriar el interior, permitiendo el paso del aire exterior y renovando el que haya en el interior de la estructura. Otra ventaja de renovar el aire es que permite controlar la concentración de algunos componentes de este, como lo son el oxígeno y el CO_2 entre otros.
- El armazón debe aguantar los efectos del clima, donde se incluyen ráfagas de viento y lluvias principalmente. La forma del tejado se diseña para evitar que el agua de las lluvias quede estancada, aumentando el peso de la cubierta y destruyéndolo.
- Para el control de los sistemas se necesita de equipo y mecanismos que deben poder entrar en el invernadero, dejando suficiente espacio para la movilidad de un operario en el interior sin dañar los cultivos.

Además de la funcionalidad de la instalación, se puede analizar también el impacto económico, puesto que no deja de haber una finalidad monetaria. El tipo de vegetales y la posición geográfica donde se hace la instalación también influyen. Una planta con unas condiciones de desarrollo muy específicas creciendo en una localización de temperaturas y clima extremo necesita de un mayor equipo y, por tanto, de un mayor costo de instalación.

3. Justificación

El trabajo descrito en este documento se realiza con la intención de desarrollar las diferentes dotes del autor para la finalización de los estudios del grado de Ingeniería Electrónica Industrial y Automática. Algunas de las tareas que aportan al crecimiento del autor en el ámbito de la ingeniería son las siguientes:

- Trabajar en las diferentes fases de diseño de un proyecto de ingeniería basado en la automatización.
- Investigar diferentes campos de conocimiento para su correcta conexión con la automatización.

- Realizar cálculos y diseños para la elección y el correcto montaje de los componentes adecuados.
- Comprobar el correcto funcionamiento de un sistema de control real.

4. Planteamiento de soluciones

En la industria existen diversas opciones con las que trabajar para el diseño de un invernadero funcional. Los apartados en los que se puede dividir el proyecto son:

- Una plataforma computacional, el núcleo que controla del sistema.
- Un HMI (Human Machine Interface), una interfaz gráfica para la monitorización del sistema.
- Sensores y actuadores, que permitan la medición de las magnitudes a controlar y la ejecución de las acciones pertinentes para mantener un funcionamiento óptimo.
- Una estructura exterior, que contenga los cultivos y permita la instalación del equipo necesario.

Es la unión de estas secciones, lo que constituye la imagen final del sistema (Figura 1).

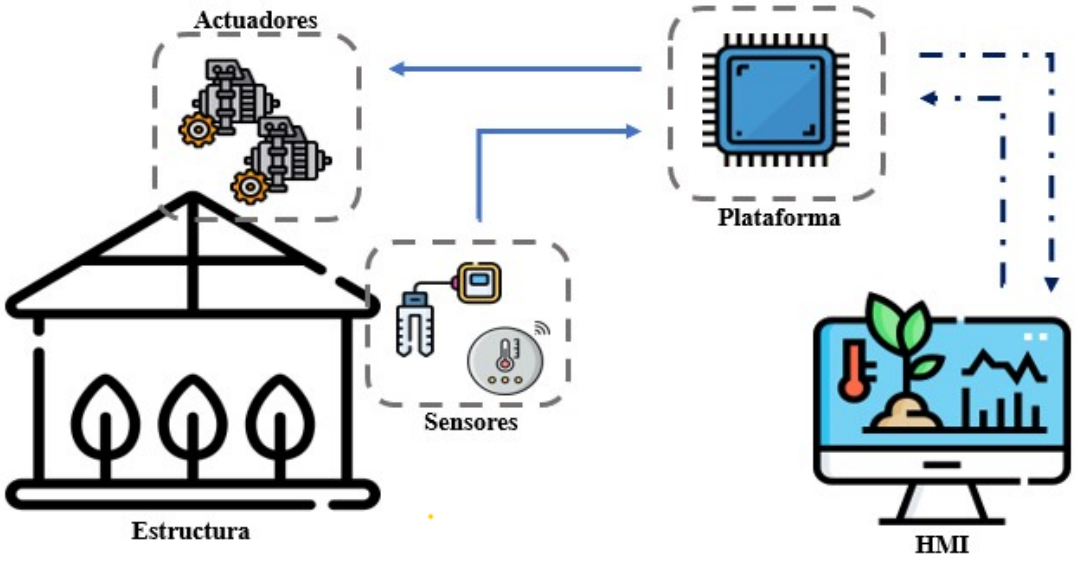


***Figura 1** - Esquema de los subapartados del sistema final. Los sensores envían datos a la plataforma que a su vez actúa mediante los actuadores. Al mismo tiempo hay información fluyendo del HMI a la plataforma en ambos sentidos.*

Se van a exponer algunas de las diferentes opciones que existen en el mercado, comparando sus características y limitaciones.

4.1. Plataforma

Esquema de los subapartados del sistema final. Los sensores envían datos a la plataforma que a su vez actúa mediante los actuadores. Al mismo tiempo hay información fluyendo del HMI a la plataforma en ambos sentidos.

La automatización del invernadero necesita una plataforma como base, a partir de la cual construir el sistema. Para este proyecto se necesita de una plataforma que pueda recibir y enviar datos,

mediante cualquier tipo de comunicación al HMI. Además, debe tener compatibilidad con los sensores y actuadores escogidos más adelante.

Se pueden dividir las opciones, de manera global, en únicamente dos:

- Plataformas basadas en una arquitectura PIC (Programmable Integrated Circuit), microcontroladores con una gran libertad de programación (Figura 2).

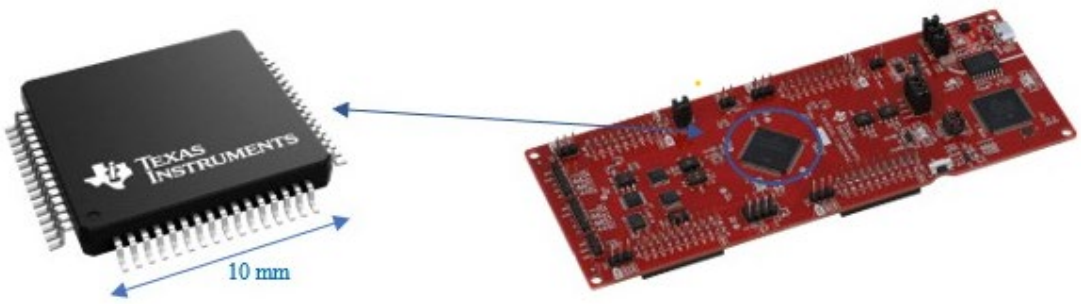


Figura 2 *– Dimensiones de un F280049C, de la marca Texas Instruments. Se indica su posición en un circuito embebido, en este caso una placa de desarrollo creada para la facilidad del uso del microchip en cuestión.*

- Plataformas basadas en una arquitectura PLC (Programmable Logic Controller), procesadores enfocados principalmente a la industria (Figura 3).

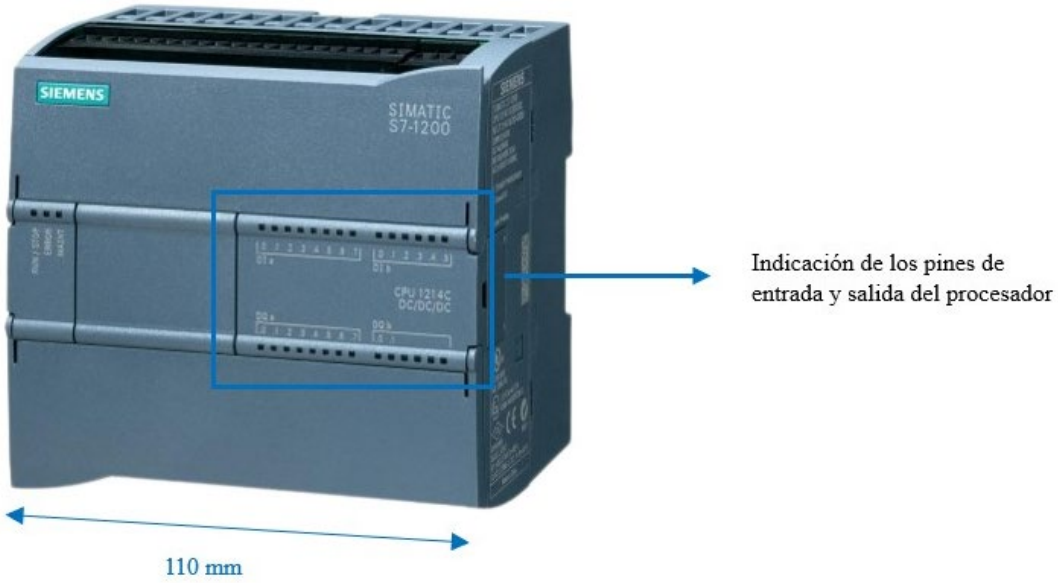


Figura 3 *– PLC de la marca Siemens, modelo 6ES7214-1AG40-0XB0. Se indica una de sus dimensiones a modo de referencia y la distribución de los pines de salida y entrada con los que cuenta el modelo.*

Los PLC son controladores, fáciles de programar y de una construcción muy fiable. Son más resistentes al ruido y a perturbaciones que otros dispositivos y algunos modelos cuentan con la posibilidad de ampliación de sus capacidades. Además, cuentan con salidas a relé, que permite el uso de actuadores de gran consumo, de ahí su amplia utilización en el ámbito industrial.

Existen varias marcas con sus respectivos modelos, cada uno con diferentes características, como la cantidad de entradas y salidas, la memoria de datos y la alimentación del dispositivo (Tabla 1).

Tabla 1 *– Tabla comparativa de algunos de los modelos de las marcas más utilizados de PLCs. Se comparan el número de entradas y salidas, así como su propósito, además de la memoria de código, alimentación a la que funcionan y un precio representativo.*

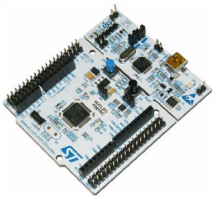
	Marca y modelo	Entradas y Salidas	Memoria	Alimentación	Precio
	Siemens Simatic S7-1200 6ES7212-1BE40-0XB0	8 entradas digitales 6 salidas a relé 2 entradas analógicas	75 kB	230 V Nominales [85 - 264] V Admisibles Corriente alterna	300 €
	Allen Bradley Micro 850 2080-LC50-24QWB	14 entradas digitales 10 Salidas a relé 4 Salidas de alta velocidad	30 kB	24 V Nominales Corriente continua	344 €
	Omron CP-Series CP1L-EM40DT	24 entradas digitales 16 salidas digitales 2 entradas analógicas	42 kB	24 V Nominales Corriente continua	500 €
	Schneider Electric Modicom M221 TM221C16R	16 entradas/salidas digitales 7 salidas a relé 2 Entradas analógicas	256 kB	[100 - 240] V Nominales [85 - 264] V Admisibles Corriente alterna	201 €

Por otro lado, las plataformas basadas en arquitectura PIC genera más diferencias en cuanto al entorno de programación con el que cuenta cada una. A diferencia de los PLC, estos procesadores están más pensadas para aplicaciones de menor potencia, con montajes más compactos, pero con una mayor personalización.

Los diferentes modelos de cada plataforma tienen velocidades de ejecución distintas, así como las capacidades de memoria, el número de entradas y salidas, los protocolos de comunicación que usan y algunas otras características específicas de cada marca (Tabla 2).

Una gran diferencia entre las plataformas basadas en PIC o PLC es el precio que tienen, debido al ámbito al que están dirigidos. Sin embargo, los microcontroladores pueden también ejecutar su función en casi cualquier ámbito, como en el diseño de un invernadero automatizado. El mayor inconveniente es la corriente máxima que puede dar en sus salidas, impidiendo una conexión directa de algunos actuadores. Sin embargo, esto se puede suplir con un circuito externo controlado por el microcontrolador.

***Tabla 2** – Tabla comparativa de algunos modelos de controladores montados en una placa de desarrollo que permite una fácil utilización del componente. Se comparan algunos datos relevantes como las entradas y salidas con las que cuentas, los protocolos de comunicación que pueden usar, la alimentación que necesitan y un precio orientativo.*

	Modelo	Características	Alimentación	Precio
	Arduino Arduino Mega 2560	54 I/O Digitales 16 Entradas analógicas Sistema ADC y DAC integrado Comunicación UART, I2C y SPI 256 kB de memoria	5 V Nominales Hasta 800 mA Regula hasta [7 - 12] V	49 €
	Raspberry Pi Raspberry Pi 4 Model B	28 pines GPIO Conexiones USB, HDMI y Ethernet Comunicación UART, I2C y SPI Protocolos Wifi y Bluetooth Hasta 8 GB de SRAM	5 V Nominales [2.5 - 3] A	90 €
	Espressif Systems ESP32	30 pines GPIO Sistema ADC y DAC integrado Comunicación UART, I2C y SPI Protocolos Wifi y Bluetooth 520 kB de memoria SRAM	5 V Nominales	12 €
	STMicroelectronics NUCLE-F091RC	64 pines GPIO Sistema ADC y DAC integrado Comunicación UART, I2C y SPI 256 kB de memoria	5 V Nominales Regula hasta [7 - 12] V	14 €

4.2. Actuadores y sensores

Se pretende que el sistema sea capaz de hacer mediciones sobre las condiciones que permiten el correcto desarrollo de los cultivos, así como tomar medidas para el control de estas magnitudes mediante diferentes actuadores. Las magnitudes que se pretenden monitorizar con el sistema son las necesidades básicas de una planta:

- Humedad. Se pretende controlar el riego mediante una bomba de agua, además de utilizar válvulas para controlar el flujo de agua en el sistema. Para medir la humedad en el aire del interior del invernadero o la humedad en el suelo de las plantaciones se utiliza un higrómetro.
- Temperatura. Mediante un termómetro se puede monitorizar la temperatura dentro de la estructura del invernadero. Para controlar la temperatura se usan calefactores y ventiladores para subir o bajar la temperatura del aire en función de las necesidades específicas.

- Luminosidad. Las horas de luz que recibe una planta pueden no ser suficientes con la incidencia del sol, para lo que se usará iluminación artificial mediante leds. Sin embargo, puede ser el caso contrario, un exceso de horas de luz, que se puede evitar con una cubierta controlada mecánicamente. Para medir la luminosidad exacta que reciben los cultivos se puede utilizar un luxómetro. También se puede utilizar un fotorresistor regulado a cierta cantidad lumínica para contabilizar el tiempo de luz solar.

Al actuar sobre estas tres variables con los diferentes componentes como entradas y salidas del controlador, se logra el funcionamiento óptimo del sistema (Figura 4).

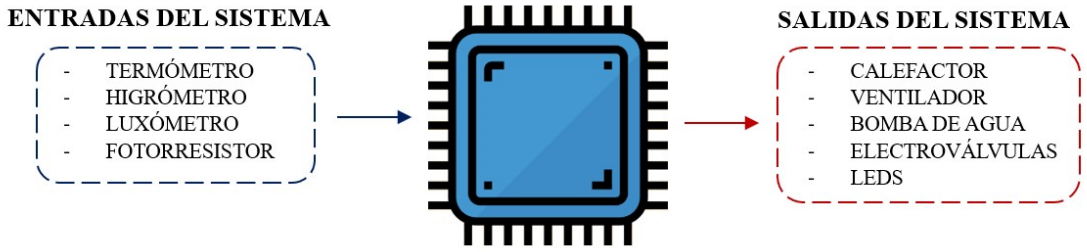


***Figura 4** – Diagrama de bloques con los diferentes componentes de entradas y salidas del sistema. Se indica la dirección que tiene el flujo de información, de sensores a procesador y de este a los actuadores.*

Los sensores tienen características para tener en cuenta a la hora de implementarlos en un diseño. La tensión de alimentación, la sensibilidad, el rango de medida y la precisión de la medición son las características más importantes (Tabla 3).

***Tabla 3** – Tabla comparativa de algunos modelos de sensores de humedad, temperatura e intensidad lumínica. Se muestran datos sobre las mediciones que pueden proporcionar . así como un precio orientativo.*

Fabricante	Modelo de sensor	Características de medida	Alimentación	Precio
MULTICOMP	HCZ-D5	Medida: Humedad relativa (HR) Rango de medida: [20 - 90] % HR Precisión: ±5 % HR	1 Vrms 0,2 mW	4,20 €
TITEC	HTFB2	Medida: Temperatura en líquido (ºC) Rango: [(-50) -180] ºC Precisión: ±0,3 %	[15 - 35] V	80,00 €
LUTRON	LX-101	Medida: Intensidad lumínica (lux) [0 - 50000] lux Precisión: ±5 % Tiempo de muestreo: 0,4 s	9 V 2 mA	80,00 €

Los diferentes modelos de actuadores tienen, al igual que los sensores, características eléctricas variadas, además de sus propias limitaciones sobre en qué grado pueden manipular la magnitud a la que están asignados (Tabla 4).

Tabla 4 *– Tabla informativa sobre algunos de los componentes actuadores posibles para el control de un invernadero. Se indica tanto modelo como fabricante, así como una tensión de alimentación y una breve descripción de cada componente.*

Fabricante	Modelo	Descripción	Alimentación	Precio
STERWINS	Bomba de superficie 900 W	Bomba monocelular con caudal máximo de 3800 l/h a una presión de 4,3 bar.	230 V AC	80,00 €
JP FLUID CONTROL	Válvula ST-SA	Válvula solenoide de 2 vías. Máxima presión de 10 bar. Caudal de 24,84 l/min a una presión de 1 bar.	24 V DC	52,80 €
SHOWGEAR	Ventilador SF-125	Ventilador con un flujo de aire 2489 m3/h.	240 V AC	145,00 €
ORBEGOZO	Calefactor cerámico FHR 3050	Calefactor cerámico con una potencia calorífica máxima de 3 kW.	230 V AC	59,00 €
VALOYA	Lámpara HPS Solray385 BX120	Lámpara de tecnología HPS para invernaderos. Intensidad luminosa máxima de 2.5 µmol/W.	230 V AC	480,00 €

Los criterios de diseño van a limitar las opciones, teniendo en cuenta las características mencionadas. Sin embargo, algo que facilita el proyecto es usar componentes con características eléctricas similares, igualando las entradas y salidas del procesador sin necesitar de un circuito externo para cada componente.

4.3. Estructura externa

El exterior de un invernadero se divide en dos partes, el armazón y la cubierta. Se ha dicho ya que la estructura busca estabilidad y resistencia, sin embargo, tienen otras características ya que están pensados en la economía de su instalación.

Generalmente el montaje se apoya en pivotes introducidos en el terreno, pero sin ocupar una gran superficie, aprovechando así al máximo la tierra para plantar. Además, se ayudan de anclajes exteriores y otros elementos de construcción para reforzar la estructura, sin perder la ligereza del conjunto.

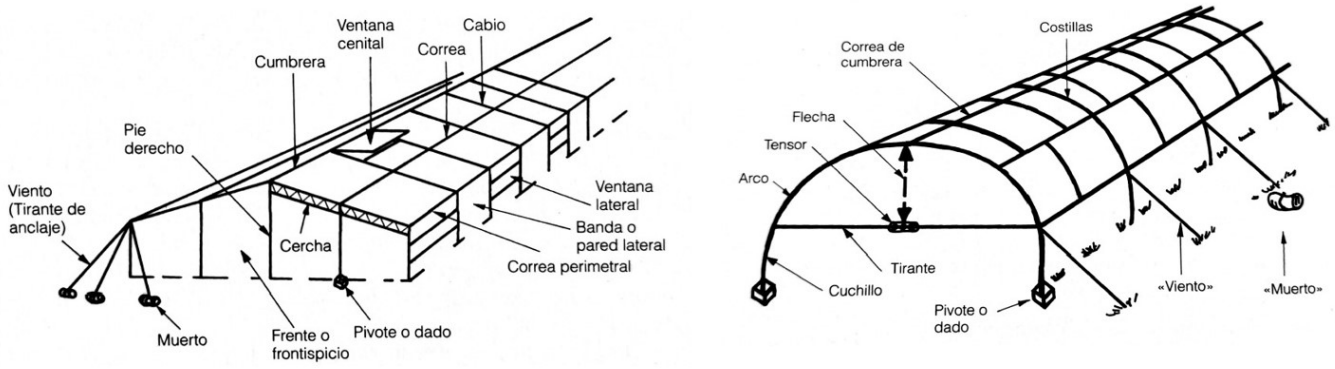


Figura 5 *– Dos tipos de estructuras, ambas de tipo capilla. La figura de la izquierda representa una de tipo capilla a dos aguas, mientras que la de la derecha es de tipo capilla con tejado curvado. Se indican también algunos elementos constructivos sobre los que se basa su arquitectura.*

Existen una variedad de diseños estructurales que difieren principalmente en la forma del tejado y la distribución interior de vigas y elementos constructivos para asegurar y distribuir la masa a los pilares principales, llamados pie derecho (Figura 5).

Todas las estructuras tienen un tejado que evita la acumulación de agua y nieve en caso de haber lluvias o un temporal frío. También ayudan a la circulación natural del aire, instalando ventanas en las zonas altas y aprovechando la tendencia al aire caliente a ascender.

Por otro lado, es importante especificar los materiales de los que está hecho el armazón, ya que es la parte que deberá soportar las diferentes cargas de peso de toda la instalación. El material que mejores prestaciones da para el esqueleto es el hierro, contando con diferentes formas de comercialización. Se pueden encontrar tubos de hierro, hierro galvanizado, hierro laminado o hierro platinado. Este material, pese a aportar gran estabilidad, también tienen una mayor densidad y, sobre todo, un mayor coste.

Otros elementos que se pueden encontrar son aluminio y madera. Estos materiales son más baratos, pero no se suelen utilizan para componer la estructura entera. Tienen un menor peso que el hierro, por lo que hacen bien su función de elementos de apoyo en la estructura interna. Aun así, se puede hacer una construcción entera más económica de madera, con menores prestaciones, pero igual de funcional. Existen diferentes tipos de maderas, dependiendo del tipo del árbol del que vengan o de la composición artificial que se haya creado de este material. Entre estas composiciones se destaca la madera laminada, o CLT.

En cuanto a la cubierta, debe cumplir con dos características principales, ser aislante térmico y dejar pasar la radiación solar. De igual manera, se pueden estudiar otras características del material:

- Reflexión de la luz.
- Absorción de la luz.
- Difusión de la luz.
- Retención de temperatura.
- Resistencia.
- Peso.

De entre los materiales más utilizados se puede hablar del plástico y del vidrio. Existen diferentes compuestos de plásticos, como lo son poliamidas, copolímero EVA, plástico "Tri-capa", PVC o policarbonato entre otras. En cuanto al vidrio, existen también diferentes compuestos, sin embargo, el más utilizado es el vidrio impreso.

La cubierta es una carga más que debe soportar el armazón, por lo que los elementos con mayor peso requieren de una estructura más estable. Por último, cabe destacar el mayor inconveniente del vidrio frente al plástico, la dificultad de montaje y su fragilidad.

4.4. Interfaz gráfica

Una vez hecha la instalación, es conveniente poder controlar la instalación sin tener que reprogramar o desmontar el núcleo del sistema, el procesador. Para la monitorización se plantea hacer uso de una interfaz gráfica, un HMI. Para la creación de este componente se deben tener en cuenta dos factores, la comunicación procesador-HMI y el dispositivo en el que se quiera usar la interfaz gráfica.

En primer lugar, existen varios métodos de comunicación, pero una división general podría ser la de comunicación alámbrica y comunicación inalámbrica. En ambos casos la comunicación se rige por un protocolo, que son las normas estándares mediante los que se comunican dos dispositivos. De esta manera, se logra una transmisión de información y de datos que es usada de la manera correspondiente, en caso de este proyecto, la representación visual de los procesos llevados a cabo por el sistema.

De entre todos los protocolos de comunicación, se destacan los siguientes:

- Modbus RTU, comunicación alámbrica basada en el envío de datos mediante solicitud. Existen dos dispositivos, el maestro que solicita la información y un mínimo de un esclavo que envía los datos. Se trata de una de las comunicaciones más comunes en el ámbito industrial [2].
- I2C, comunicación alámbrica basada en el envío de datos mediante una trama de bits. Se utiliza comúnmente para el envío de información de sensores, actuando como esclavo, a un controlador, actuando como maestro.
- UART, comunicación alámbrica basada en la transmisión de información usando una trama de bits. Se trata de una comunicación entre dos dispositivos, con la peculiaridad de que solo se puede enviar información en un sentido a la vez.
- Wi-Fi 802.11.x, comunicación inalámbrica basada en la conexión a puertos. Como peculiaridad, permite una conexión segura mediante autenticación.
- WirelessHart, comunicación inalámbrica diseñada para el uso de varios dispositivos que forman una red. La transmisión de datos de basa en la comunicación por radio ISM [3].

En segundo lugar, se encuentra el lenguaje de programación, que dictará a la vez los dispositivos en los que se pueda utilizar el programa creado. El programa creado puede estar orientado a ser una aplicación de escritorio, compatible con algún sistema operativo como Windows, Linux o macOS. De igual manera, puede ser orientado a ser utilizado en una aplicación móvil o incluso a ser utilizado a través de internet, siendo una aplicación de la nube.

Cada sistema operativo tiene su propia interfaz de programación de aplicaciones, o API, así como un lenguaje de programación en el que se basa. En el caso de Windows, se utiliza Windows Forms, una API basada en el lenguaje C#. Por otro lado, macOS cuenta con Cocoa, que a su vez usa el lenguaje de programación de Swift.

En cualquiera de los casos, la función del programa es la misma, la de comunicarse con el procesador que controla el invernadero. Por esto, independientemente del entorno de desarrollo, lo importante es la coherencia que tenga a la hora de hacer uso del protocolo de comunicación elegido.

5. Descripción detallada de la solución adoptada

Para lograr diseñar un invernadero funcional, cualquiera de las opciones mencionadas es adecuada. Sin embargo, se pretende reducir los costes lo posible sin perder calidad en el funcionamiento del sistema.

5.1. Arduino

De entre las plataformas basadas en microchip, Arduino se destaca por su facilidad de uso, los componentes dedicados a este procesador y la cantidad de soporte que existe en internet para su uso. Sus modelos cuentan con una capacidad de procesamiento más que suficiente, teniendo en cuenta que el sistema que debe controlar no tiene un volumen de datos muy grande, ni tampoco necesita de una velocidad de actuación instantánea.

De entre los modelos con los que cuenta Arduino, se va a usar el Arduino NANO (Figura 6), que cuenta con las siguientes características:

- Frecuencia de reloj de 16 MHz.
- 22 pines digitales configurables tanto de entradas como de salidas.
- 8 pines de entrada analógicos configurables para funcionar como salidas digitales.
- 6 canales de salida PWM.
- 32 kB de memoria.
- Conexión serial USART.
- 5 V de alimentación
- Consumo máximo de 800 mA.
- Pines de salida con una corriente máxima de 20 mA.

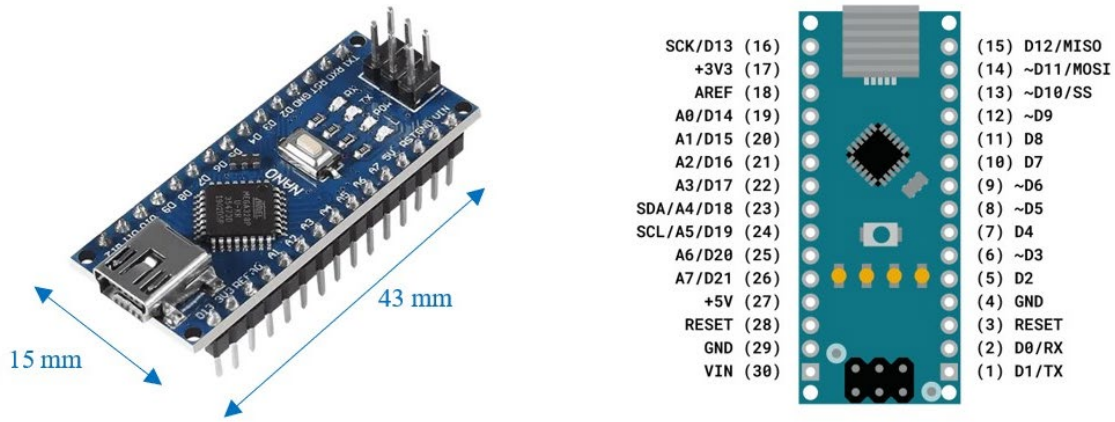


Figura 6 *– Modelo NANO de la marca Arduino, indicando sus dimensiones. Se incluye también la numeración y nomenclatura de los pines con los que cuenta.*

El número de pines de entrada y salida son suficientes para manejar los componentes necesarios para un invernadero. Cabe destacar, que incluso se podrían añadir más sensores o actuadores para ampliar la instalación o mejorar la monitorización de alguna de las variables a controlar.

Esta plataforma cuenta con diferentes protocolos de comunicación, y este modelo en concreto cuenta con la posibilidad de una comunicación serial UART, que será vital para la transmisión de datos entre el procesador y el HMI.

Por otro lado, el mayor inconveniente que puede tener este procesador es la corriente que puede proporcionar en sus pines de salida, donde irían conectados los actuadores. Como se verá más adelante, la potencia consumida por estos componentes es superior a la que puede manejar el Arduino NANO. Sin embargo, esto se puede solucionar con una alimentación externa para estos dispositivos, controlando el circuito con la salida del procesador mediante un transistor BJT o un relé.

5.2. Actuadores y sensores

Como se ha mencionado antes, para hacer más sencillo el diseño, se van a elegir componentes con una tensión de alimentación similar. Gracias a esto, en el circuito final solo hará falta una alimentación externa con la que alimentar todos los componentes que no pueda soportar el Arduino NANO.

En primer lugar, los sensores elegidos tienen una gran compatibilidad con la plataforma de Arduino:

- DHT11 (Figura 7). Se trata de un sensor alimentado a 5 V, con un consumo máximo de 2.5 mA, que proporciona medidas de temperatura y humedad mediante un pin digital. Pese a ser un pin digital envía una señal analógica codificada a una trama de datos que el propio Arduino transforma a un valor analógico mediante las bibliotecas correspondientes. Este sensor puede medir la temperatura en un rango de [0 - 50] ºC, además de la humedad relativa del aire en un rango de [20 – 90] %HR. Su hoja de datos nos proporciona también la precisión, de ±5 %HR y ±2 ºC, así como su resolución, de 0.1 ºC y 1 %HR [4].
- FC-28 (Figura 7). Para medir la humedad de la tierra se va a usar este sensor, conectado a su vez con un pequeño módulo comparador basado en un LM393, un amplificador operacional. Este módulo es alimentado a 5 V y permite dos tipos de salidas, una digital que compara la humedad medida con un límite programable, y una salida analógica que proporciona un valor entre 0 y 1023, correspondiente al rango de medida del sensor de [0-100] %HR. La hoja de datos del componente no indica la corriente de consumo por lo que analizará posteriormente para comprobar que no supone un problema para el circuito [5].
- GL5539 (Figura 7). Un fotorresistor conectado a un módulo comparativo LM393 al igual que el higrómetro anterior. La alimentación es misma que en el caso anterior, con la falta de corriente que se medirá posteriormente de manera empírica. El fotorresistor, o LDR, se pretende usar para medir si existe luz o no, y el límite que medirá será también regulado mediante el LM393, proporcionando una salida digital de 1 o 0.

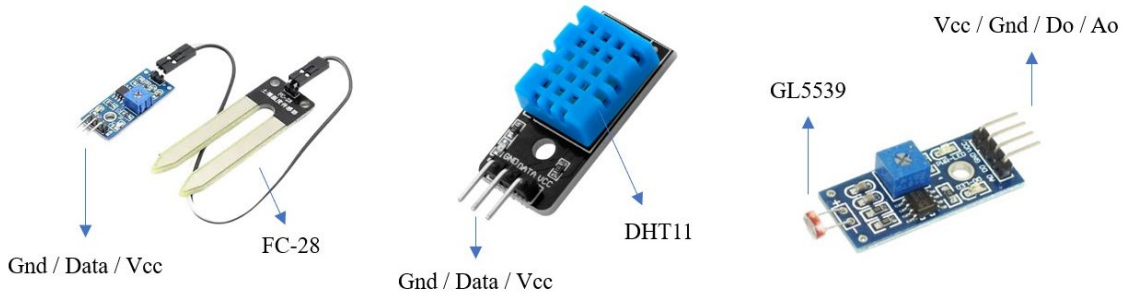


***Figura** 7 – A la izquierda se encuentra el FC28 conectado al módulo LM393. Se indican los pines de conexión que tiene el mismo para su conexión con el procesador. En la figura del centro está el DHT11, montado sobre una placa con una resistencia protectora del componente. Se indican también los pines de conexión que tiene, siendo el pin central por donde envía su trama de información. A la derecha se encuentra otro módulo LM393 con sus respectivas conexiones, además del componente que regula, el fotorresistor GL5539.*

Para una mayor precisión, puede que sean necesarios varios sensores de cada modelo, para medir correctamente proporciones de espacio de un invernadero. Sin embargo, para una maqueta a escala, un sensor de cada tipo puede ser suficiente para un control correcto.

En segundo lugar, los actuadores, donde se va a diferenciar los componentes usados para un modelo a escala real y los componentes usados para una maqueta a escala reducida.

La bomba de agua debe ser capaz de proveer una cantidad de agua mínima, teniendo en cuenta el tamaño del cultivo y las necesidades hídricas de los vegetales. Para calcular las necesidades hídricas se va a hacer uso de una herramienta oficial del gobierno de España [6], que proporciona un cálculo de la cantidad de agua necesaria por metro cuadrado (ETc-Pe(mm)) de un cultivo determinado en una ubicación del territorio español (Figura 8). El cálculo, de manera simplificada, utiliza la evapotranspiración de otra especie como base (ET_0) y un coeficiente que depende de las condiciones a las que crece la planta deseada (K_c).

$$ET = ET_0 \cdot K_C$$

El valor resultante (ET), es la evapotranspiración de la planta a cultivar, un parámetro que mide la cantidad de agua que pierde el vegetal de manera natural, evaporándola en sus procesos de crecimiento. Por tanto, es una medida que sirve también para conocer la cantidad de agua que se le debe proporcionar para mantener unas condiciones óptimas.

Estación: Algemesí (Valencia/València, Valencia)

Comarca: Riberas del Jucar

Cultivo: Otras flores

Fecha inicial: 01/04/2023 **Fecha final:** 11/07/2023

Volcar el listado a un archivo CSV

Fecha	Kc	ETo(mm)	Etc(mm)	Pe(mm)	ETc-Pe(mm)
abr./2023	0,60	122,76	73,65	0,00	73,65
may./2023	0,60	122,25	73,35	22,52	50,83
jun./2023	0,80	138,46	110,77	10,17	100,60
jul./2023	0,90	48,25	43,43	8,80	34,63

Figura 8 *– Tabla de resultados que proporciona la herramienta del gobierno sobre la necesidad hídrica. En este caso devuelve valores mensuales dentro del rango de tiempo indicado, en una región del territorio español.*

En este caso, ninguna de las especies que proporciona la herramienta se va a cultivar, por lo que se usa como cultivo la opción de "Otras flores". La herramienta permite calcular el valor de evapotranspiración en función de los días, semanas o meses seleccionados. El valor proporcionado se puede leer como los milímetros de agua que llena de un cubo de 1 m^3, equivalente a un litro por milímetro llenado. Este valor se va a sobredimensionar para tenerlo como referencia en la elección de la bomba de agua.

Las características de las bombas de agua indican la cantidad de agua que pueden proporcionar por hora, por lo que se hace el cálculo para el cambio de unidad, cogiendo el valor máximo:

$$ET = 100.6 \frac{l}{mes}$$

$$ET = 100.6 \frac{l}{mes} \cdot \frac{1\,mes}{30\,dias} = 3.36 \frac{l}{dia}$$

$$ET = 3.36 \frac{l}{dia} \cdot \frac{1\,dia}{24\,h} = 0.14 \frac{l}{h}$$

Este valor es la cantidad de agua que se necesita por metro cuadrado, por lo que suponiendo un área de cultivo (A) como referencia de 30 m^2:

$$ET = 0.14 \frac{l}{h} . A$$

$$ET = 0.14 \frac{l}{h} \cdot 30\,m^2 = 4.2 \frac{l \cdot m^2}{h}$$

La bomba escogida puede proporcionar una cantidad de agua superior al cálculo requerido, y trabaja en una tensión de 12 V con un consumo máximo de 8 A (Figura 9) [7].

Figura 9 *– Bomba de la marca de SEAFLO, en concreto el modelo SEAFLO 33 Series DC Diaphragm Pump de 12 V. Se indican las dimensiones del componente además de la entrada y salida de agua que tiene.*

El sistema de riego no está completo sin las respectivas electroválvulas, que deberán ser alimentadas con la misma tensión que la bomba, 12 V (Figura 10). El fabricante proporciona también los datos de consumo, 500 mA, además de otros datos como su tiempo de respuesta [8].

Figura 10 *– Imagen de una electroválvula de la marca Velleman. Se indican sus bornes de conexión, así como el diámetro que tienen el orificio de salida y entrada de agua del componente. En el orificio izquierdo está instalada una boquilla adaptadora que reduce el tamaño de entrada.*

Continuando con los actuadores, para el control de temperatura se va a disponer de un calefactor, o de más de uno si el tamaño del invernadero lo requiere. El modelo escogido funciona, al igual que el resto de los componentes, a 12 V, además de tener un consumo medio de unos 16 A (Figura 11). Este componente tiene integrado un ventilador, que ayuda a la distribución del aire por todo el invernadero.

Por otro lado, para enfriar la estancia y ayudar con la renovación de aire, se usará un ventilador, que impulse el aire caliente hacia la parte superior del invernadero. Lo ideal sería instalar los

Figura 11 *– A la izquierda se puede ver la imagen de un calefactor y su cable de conexión para alimentarlo a 12 V. Se observa también la medida de largo del componente. A la derecha está la imagen de un ventilador que funciona a 12 V y tiene un diámetro de 20 cm.*

ventiladores en las entradas de aire, delante de las ventanas. Estos ventiladores funcionan con una tensión nominal de 12 V, con un consumo de 200 mA (Figura 11).

Como se ha comentado ya, la estructura del invernadero deberá tener ventanas, que pueden estar controladas de manera automática mediante un motor eléctrico y un sistema mecánico como pueden ser mediante engranajes, poleas o cremallera. Se elige un motor eléctrico conectado a un reductor, un motorreductor de 12 V (Figura 12). Se trata de un componente con un máximo de 5 Nm de par, diseñado específicamente para este tipo de tareas [9].

Figura 12 *– Imagen de un motorreductor, compuesto por dos partes. La parte derecha es la parte motorizada, que genera el giro con un determinado par y una determinada velocidad. La parte izquierda, la transmisión reductora, reduce la velocidad de giro aumentando el par del componente. Las condiciones finales se dan a partir del eje de giro.*

Por último, es necesario una iluminación artificial en caso de que la radiación solar no incida sobre las plantas el suficiente tiempo. Existen varios estudios sobre cómo afecta la iluminación artificial a las plantas, siendo el concepto más importante el PAR, o radiación fotosintéticamente activa [10]. Según este concepto, la longitud de onda que más favorece al crecimiento de las plantas se encuentra también en el espectro visible, el que puede captar el ojo humano. Basándose en este concepto, las longitudes de onda que más favorecen al vegetal son las correspondientes a los colores rojo y azul (Figura 13).

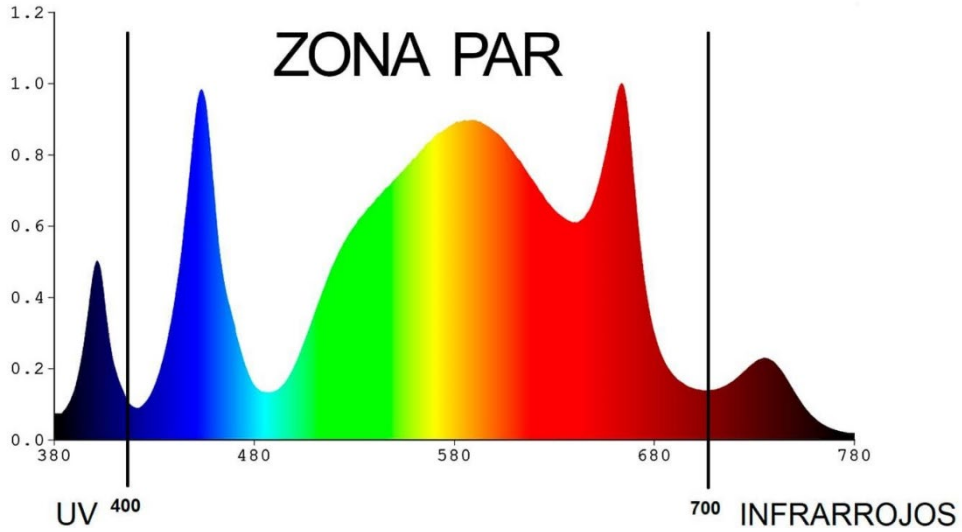


Figura 13 *– Gráfica que muestra la eficiencia fotosintética frente a la longitud de onda. En el eje X se muestra la longitud de onda, en este caso los valores comprendidos en el espectro visible, los colores que el ser humano es capaz de identificar. En el eje Y se encuentra un coeficiente que representa la eficiencia fotosintética, lo que aporta a una planta para realizar la fotosíntesis. En el eje X se encuentran marcados los valores de longitud de onda correspondientes a la luz ultravioleta (UV) y la luz infrarrojos.*

En vez de utilizar una lámpara led ya montada, se va a crear una propia mediante los leds individuales. De esta manera se puede hacer un diseño concreto para la alimentación de 12 V con la que se trabaja (V_{cc}). Dado un led rojo que trabaja con una tensión de 2 a 2.4 V (V_{Led}) (Figura 14), se calcula el máximo unidades en serie (N):

$$N = \frac{V_{cc}}{V_{Led}}$$

$$N = \frac{12\ V}{2.4\ V} = 5$$

Leds para montaje superficial (SMD) SMD leds

Cápsula transparente. Water clear lens.

REFERENCIA Y COLOR / PART NUMBER AND COLOR	TIPO DE CÁPSULA / PACKAGE	LUMINOSIDAD / LUMINOUS INTENSITY	Tº COLOR/ LONGITUD DE ONDA / CCT/ WAVELENGTH	ÁNGULO / VIEWING ANGLE	CORRIENTE DE ALIMENTACIÓN / FORWARD CURRENT	TENSIÓN DE ALIMENTACIÓN / FORWARD VOLTAGE
WW-WIS190TS-G	SMD 0603	270 mcd	5500k-7000k	140°	20 mA	3,2~4,0 VDC
WW-OR190TS-E	SMD 0603	120 mcd	630 nm	140°	20 mA	2,0~2,4 VDC
WW-BIS190TS-G	SMD 0603	125 mcd	475 nm	140°	20 mA	3,2~4,0 VDC
WW-GIS190TS-G	SMD 0603	40 mcd	525 nm	140°	20 mA	3,2~4,0 VDC
WW-YHS190TS-E	SMD 0603	80 mcd	595 nm	140°	20 mA	2,0~2,4 VDC

Figura 14 *– Hoja de características de un led SMD, en concreto el modelo SMD 0603. Se indica el color deseado, del que se proporcionan los valores de tensión y corriente de alimentación.*

Se puede observar q a una tensión máxima, los leds crean una caída de tensión igual al de la fuente, por lo que no debería hacer falta una resistencia reguladora para este caso.

Ahora bien, los actuadores nombrados hasta el momento están pensados para trabajar con una carga mayor de la que habría en un invernadero a pequeña escala. Por este motivo, se van a elegir componentes con menores características, que a su vez trabajarán con una menor tensión de alimentación. Cabe mencionar, que algunos de los siguientes componentes no proporcionan datos exactos sobre algunos de sus parámetros, por lo que los que sean necesarios se calcularán de manera experimental más adelante.

En primer lugar, la bomba de agua y las electroválvulas, ambas funcionando a 5 V (Figura 15). La bomba de agua es de vacío de presión negativa, mientras que la válvula solenoide es normalmente abierta. Se proporcionan algunos datos técnicos, pero se comprobará más adelante que pueden cumplir su función y el consumo que realmente tienen.

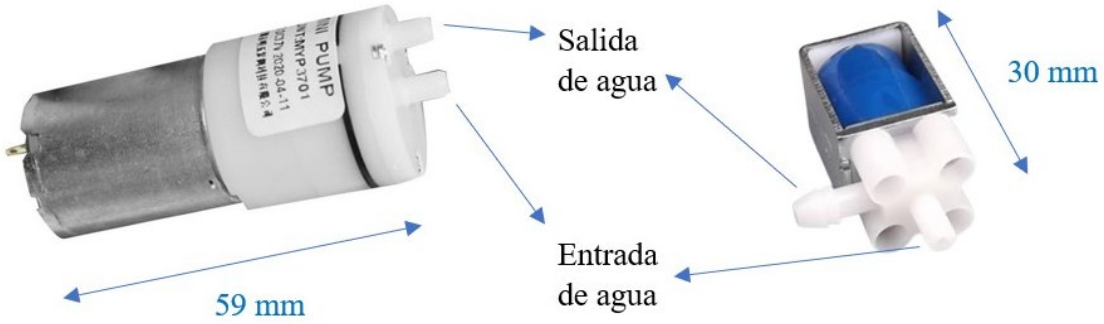


Figura 15 *– A la izquierda se encuentra una mini bomba de agua con alimentación a 5 V. A la derecha una electroválvula, un solenoide que funciona también a 5 V. Se indica en ambos componentes la medida de largo, así como la entrada y salida de agua en ambos casos.*

Para completar el sistema de riego harán falta tuberías con diámetro de las salidas estos componentes.

En segundo lugar, para el control de temperatura, Se van a utilizar dos ventiladores de 6 cm de diámetro, además de un calefactor de cerámica MCH (Figura 16). Ambos componentes funcionan a 5 V, con una potencia máxima de consumo de 0.3 y 8 W respectivamente.

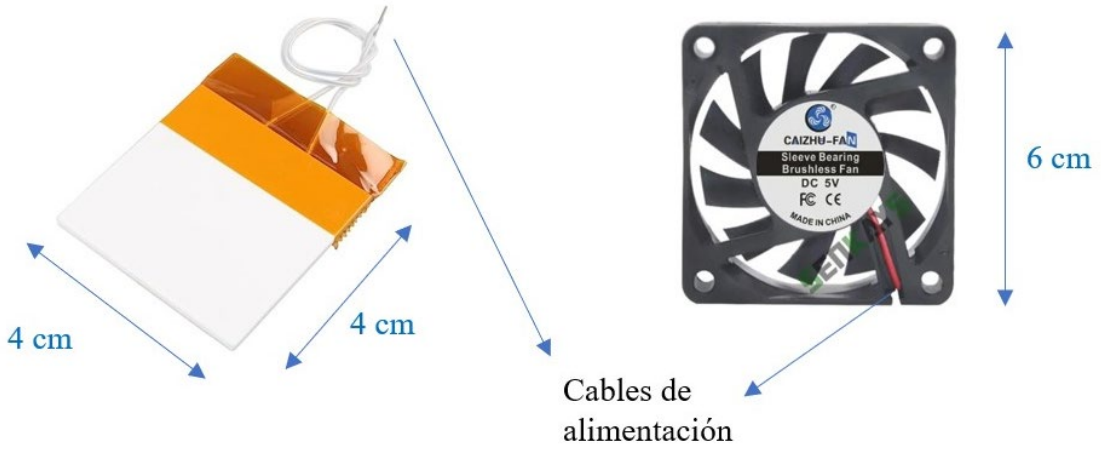


Figura 16 *– A la izquierda la imagen de un calefactor de cerámica metal, o MCH. A la derecha un ventilador construido en plástico. En ambos componentes se indican sus medidas, así como el cable de conexión que tiene para alimentación.*

En el caso de la ventilación, se pretende colocar un ventilador en la entrada de aire, y otro en la salida, dirigiendo el flujo de aire para renovar el aire del interior del invernadero. En el caso de la maqueta, no se va a utilizar un mecanismo automatizado para abrir y cerrar las ventanas, pero se intentará remediar más adelante con el diseño y montaje de la estructura.

Por otro lado, el MCH no dispone de un ventilador que distribuya el aire caliente por el interior del invernadero, sin embargo, las dimensiones del interior de este permiten que no haga falta crear un flujo de aire caliente.

Para finalizar, la iluminación artificial debe tener la misma longitud de onda que en el diseño a escala real. Por este motivo, se utilizarán los mismos leds con una distribución diferente:

$$N = \frac{V_{do}}{V_{Led}}$$

$$N = \frac{5\ V}{2.4\ V} = 2.1 = 2$$

En este caso, estos componentes tienen un consumo que, si puede proporcionar el Arduino NANO (V_{do}). Debido a esto, la conexión será directa, sin necesitar de un relé o transistor BJT. Sin embargo, a diferencia del caso con los 12 V, en este caso si se utilizará una resistencia, calculada más adelante, para proteger los componentes de una sobretensión.

5.3. Estructura y materiales

De entre los tipos de estructura propuestos anteriormente, este proyecto se va a basar en el de tipo “capilla” (Figura 17). En este modelo, que es también el más común, se pueden tener dos porciones de terreno con cultivos independientes, dejando un espacio en el centro para el paso de los trabajadores. Esta disposición permite además la expansión del invernadero, tanto en el eje x como en el eje y, mediante un montaje modular.

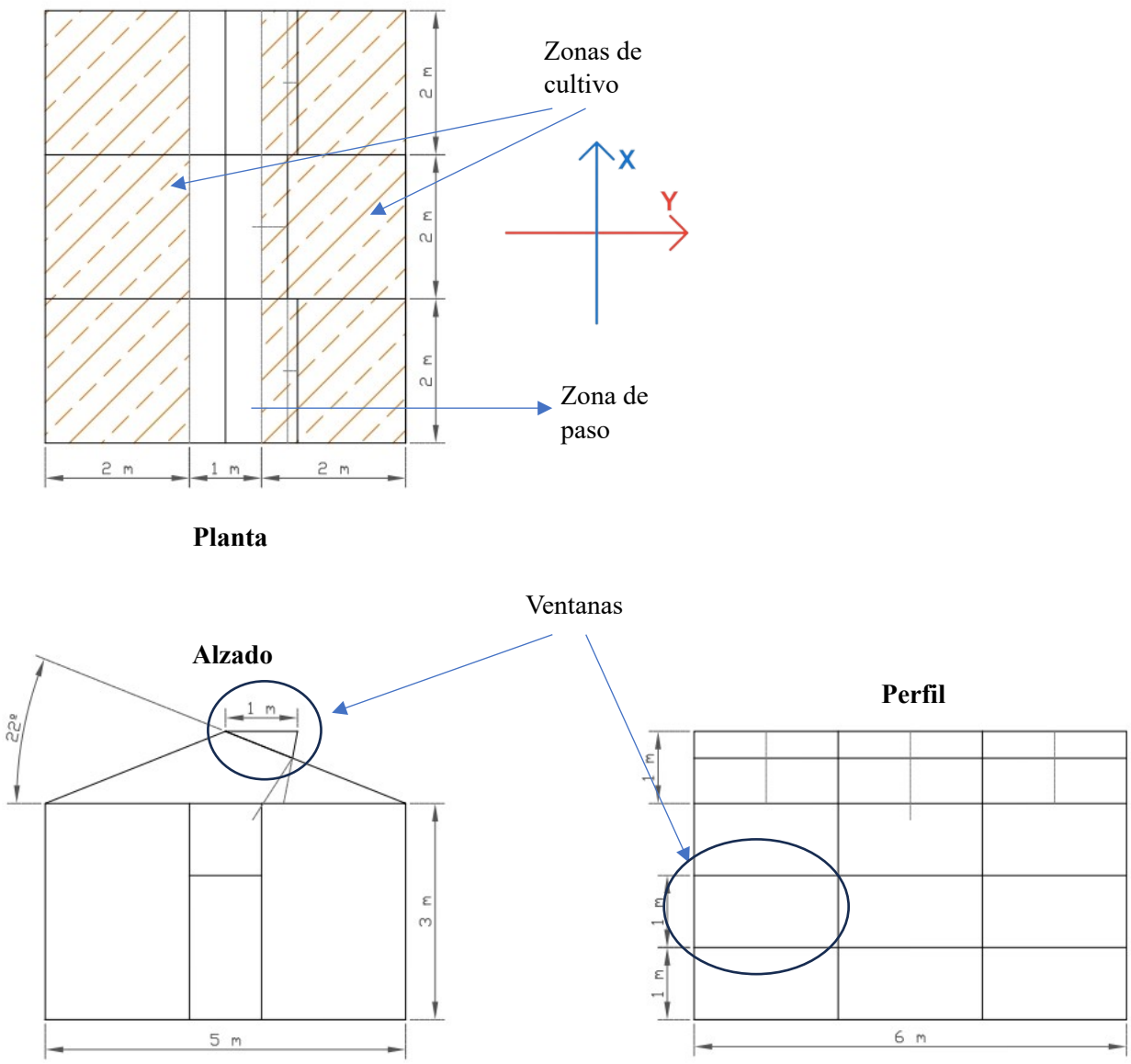


Figura 17 *– Planta, alzado y perfil del plano de un invernadero capilla a dos aguas. El tejado tiene inclinación para evacuar el agua de lluvia hacia sus dos lados. Tiene una ventana superior por la que evacua el aire caliente, además de ventanas laterales para la entrada de aire fresco. Se indican sus dimensiones exteriores, así como la inclinación que tiene el tejado. Los ejes de la figura son los correspondientes a la vista de planta.*

Las ventanas de esta estructura están dispuestas para una entrada de aire desde una zona más inferior, dejando la salida en el techo de la estancia, para eliminar el exceso de temperatura. El resto del equipo estaría situado en el interior, a excepción de la bomba de agua, apoyados en los distintos elementos de construcción del interior.

Las porciones del terreno deben permitir un correcto espacio de desarrollo para los cultivos, por lo que en cada lado se van a plantar tres hileras del vegetal, dejando un espacio de medio metro entre cada planta. De esta manera, el terreno que de cubrir el invernadero sería de 5 m de ancho por 6 m de largo. La estructura debe tener una altura total de 4 m, contando con el metro que añade el techo inclinado.

Partiendo de esta base, la maqueta que se va a montar en este proyecto tendrá una escala de 1:6 aproximadamente. Cabe mencionar igualmente que hay elementos que no pueden tener la misma escala, como es la altura, por motivos de funcionalidad. Esta maqueta no se va a montar sobre un terreno como se haría una real, sino más bien tendrá sus propias macetas o porciones de terreno en su interior. Debido a esto, a la altura que pueda crecer un vegetal plantado en este invernadero,

hay que sumarle las dimensiones de la maceta. Otro matiz es la eliminación del espacio de trabajo que había en el modelo real. En este caso, un operario no va a caminar dentro, por lo que ese espacio se reduce y se utilizará para el montaje del resto de equipos.

Con lo mencionado hasta el momento, el diseño de la maqueta tendrá unas dimensiones de 65 cm de ancho por 35 cm de largo. Además de contar con una altura total de 1 m, tras haberle añadido el tejado inclinado característico del tipo "capilla a dos aguas" (Figura 18).

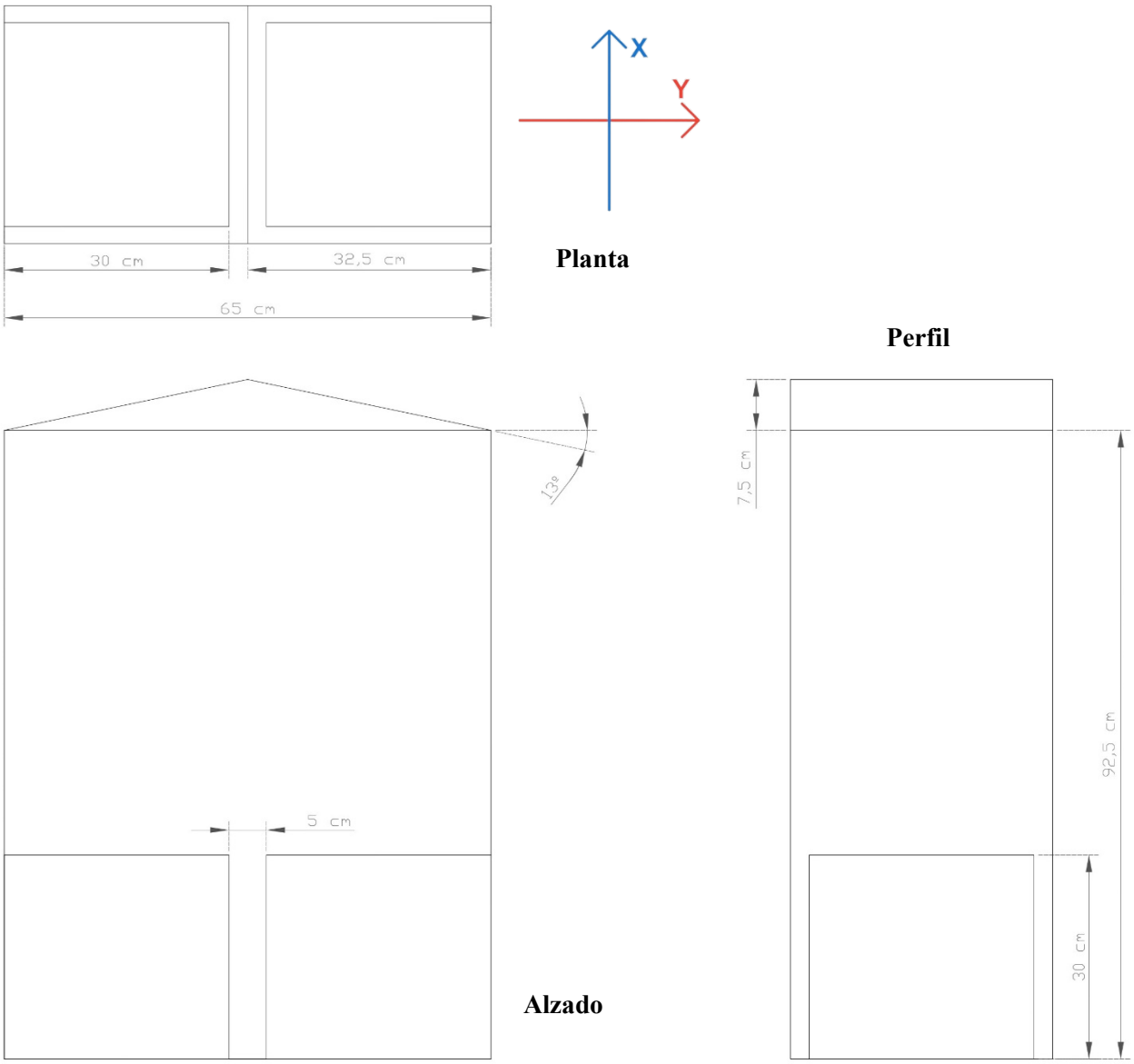


Figura 18 *– Planta alzado y perfil del plano de dimensiones de la maqueta. Se indican únicamente las dimensiones exteriores, así como la posición de las macetas en su interior. Se muestra también una cruz de ejes correspondiente la vista de planta.*

Al igual que ocurre en el modelo real, se puede pensar en una ampliación, considerando esta maqueta como un solo módulo. Así, la maqueta puede crecer en el eje x e y con módulos de invernaderos de las mismas características.

El modelo real se diseña para estar construido con tubos de hierro galvanizados. Estos tubos son una aleación que permite que sea más duradero en un ambiente húmedo como el que hay dentro del invernadero. Al ser tubos, huecos por dentro, son más ligeros, pero son lo suficientemente resistentes para soportar toda la estructura de manera estable. Además de los tubos como elemento

principal, la instalación debería tener alambres para reforzar la estructura mediante anclajes exteriores.

Por otro lado, en la maqueta se busca más la economía del montaje que la resistencia a largo plazo. De esta manera, el material con el que se planea construir la maqueta como elemento principal es la madera. Las dimensiones de la maqueta suponen una menor carga por lo que se considera que una estructura de madera puede soportar de manera estable todo el equipo necesario.

Por último, la cubierta en ambos casos será de plástico. El motivo principal es la flexibilidad del material frente al vidrio, permitiendo una instalación más cómoda y modificable.

5.4. HMI

Como último elemento se presenta la interfaz gráfica, que se planea que funcione principalmente para dispositivos con Windows. Se trata de uno de los sistemas operativos más grandes y utilizados en la actualidad, por lo que la accesibilidad al programa es más alta. Para este sistema operativo se va a programar una aplicación mediante el entorno Visual Studio, con el lenguaje C#. La aplicación final será un archivo ejecutable ".exe", que funcione en cualquier versión de Windows.

La idea de este programa es que pueda controlar los parámetros de funcionamiento del Arduino NANO como procesador principal, sin embargo, no podrá manejar los actuadores de manera manual. La función principal del HMI será la de monitorización, por lo que deberá recibir continuamente información sobre el estado de los actuadores y sensores de dentro del invernadero. El diseño gráfico de la interfaz gráfica se verá más adelante.

Para la comunicación se propone una comunicación inalámbrica para el modelo real, utilizando bluetooth para una comunicación serial basada en una trama de bits con la actualización de la información,

En el caso de la maqueta, la comunicación será serial, mediante el protocolo UART. Del mismo modo que con el modelo real, la comunicación deberá ser continua para comprobar y actualizar de manera gráfica el estado del sistema. En este caso, el traspaso de información mediante una conexión alámbrica a un puerto designado para la aplicación. Al igual que el diseño, las funciones de comunicación se explicarán más adelante.

6. Cálculo y dimensionado

Inicialmente hay que indicar el rango de valores con el que trabajar el sistema, para así, concluir el diseño a partir de estos. Así pues, se van a elegir dos tipos de plantas con condiciones de crecimiento diferentes, pero que tengan un rango común en el que ambas especies puedan convivir. Las dos plantas elegidas son las siguientes:

- Orégano. Se trata de una planta aromática que crece principalmente en la zona mediterránea. Se trata de un vegetal que requiere de una temperatura ambiente media de 20 ºC, con un rango de [15 – 25] ºC. En cuanto a la luz necesaria, requiere de un máximo de 8h, teniendo consecuencias negativas un exceso de luz. Por último, para crecer correctamente necesita de un suelo húmedo, pero no en exceso, por lo que se supondrá un rango de [45 - 60] % de humedad.
- Cilantro. Esta es una planta aromática que, de igual manera, comparte un rango de condiciones similares al del orégano. El cilantro requiere de una temperatura media de 17

ºC, teniendo un rango de [10 - 25] ºC. La humedad del suelo se fija en un rango de [45 - 60] %, ya que también necesita de un suelo húmedo continuo. Finalmente, se trata de una planta que requiere de un máximo de 6 horas de luz, límite menor que el del orégano.

Con estas condiciones se va a diseñar el sistema, sin embargo, se pretende que tenga valores modificables para permitir una mayor variedad de especies vegetales.

Cabe mencionar también que el diseño del presente documento se basa en la creación del modelo a escala, habiendo utilizado el modelo real como inspiración para este.

6.1. Código Arduino

En primer lugar, se decide un funcionamiento general que va a tener el programa, desarrollando cada subsistema más adelante (Figura 19).

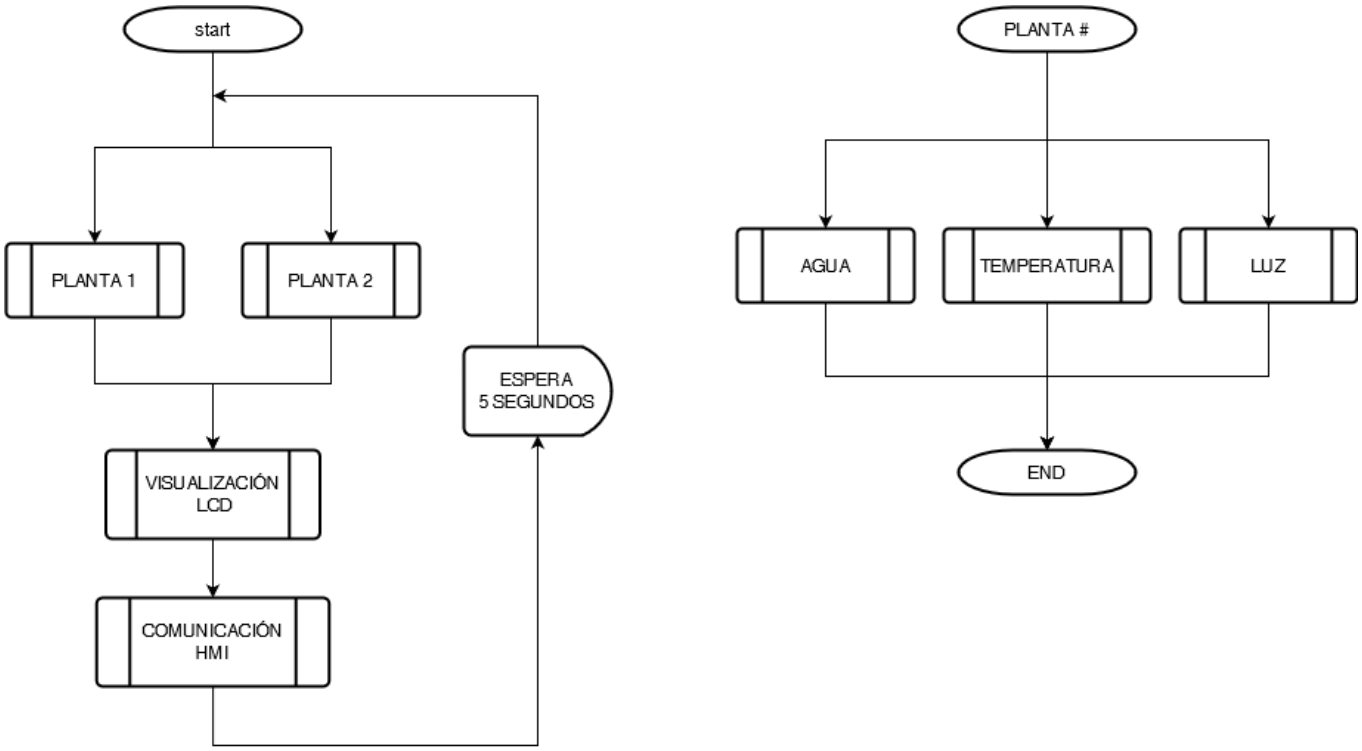


***Figura 19** - Pie de foto – Flujograma general que consta de 4 subprocesos. En el primero se comprueban las tres variables importantes de cada una de las plantas, como se muestra en el esquema de la derecha. Los siguientes dos subprocesos son de visualización para el usuario, mediante un LCD, y de comunicación con el HMI*

La idea es que el bucle principal compruebe el estado de cada una de las plantas, comparando el estado actual con el rango de valores máximos y mínimos correspondientes a las plantas cultivadas (Figura 20).

Esta comprobación se va a hacer cada 5 segundos, debido a las limitaciones del sensor DHT11, que no puede proporcionar datos sin un pequeño periodo de espera. Sin embargo, no se considera un problema, ya que este proyecto no necesita de una velocidad de actuación rápida, sino más bien, el funcionamiento continuado a lo largo de varias horas e incluso días.

```
//ComprobaciÓn del estado de las plantas cada 5 segundos
 delay (4000); //El segundo que falta está más adelante
 //Almacenamiento y conversión de datos mediante los sensores
 dhtAux = int (dht.readTemperature ());
```

```
  ldrAux = contador (digitalRead (LdrPin));
  //Planta 1
  Planta1.ContadorLuz (ldrAux); //Comprueba si hace falta más luz
artificial
  Planta1.ComprobarAgua (analogRead (Hum1Pin)); //Comprueba si hace falta
riego
  Planta1.ComprobarTemp (dhtAux); //Comprueba si la temperatura es
correcta
  //Planta 2
  Planta2.ContadorLuz (ldrAux);
  Planta2.ComprobarAgua (analogRead (Hum2Pin));
  Planta2.ComprobarTemp (dhtAux);
```

Figura 20 *– Fragmento del bucle principal del código fuente de Arduino. Este fragmento se ejecuta en cada iteración de ciclo cada 5 segundos. En este fragmento se ejecutan los métodos de comprobación de estado de cada planta con las mediciones obtenidas por los sensores.*

De cada planta se van a comprobar los tres parámetros necesarios, leyendo los valores proporcionados por los sensores de manera directa, como es el caso de "analogRead (Hum1Pin)" en la comprobación de humedad, o de manera indirecta como es el caso de los otros dos parámetros. Las funciones correspondientes a la luz y la temperatura utilizan variables auxiliares, cuyos valores han sido transformados para la correcta cohesión de tipos de variables a lo largo de todo el programa.

El "." en las funciones indica que estas instrucciones son en realidad métodos, funciones asociadas a una clase. La parte anterior al punto indica el objeto que está llamando a ese método, la parte posterior al punto, que es común a todos los objetos creados de esa clase. En este caso, se ha creado la clase "Planta", de la que se definen dos objetos al principio del programa, "Planta1" y "Planta2". La construcción de objeto se hace al momento de declarar las variables globales, mientras que su configuración inicial se hace en la función de configuración del propio programa, mediante los métodos ".SetPlanta ()" (Figura 21).

```
  //Orégano (8h, 25 grados, 15 grados mínimos, 60 % humedad max, 45
%humedad min)
  Planta1.SetPlanta (48, 25, 15, 60, 45);
  //Cilantro (6h en unidad de 10min, 25 grados máximos, 10 grados, 60 %,
45 %)
  Planta2.SetPlanta (36, 25, 10, 60, 45);
```

Figura 21 *– Fragmento de código del programa principal de Arduino. Se trata de la configuración inicial hecha para configurar los objetos asociados a la clase "Planta". En esta configuración se proporcionan los valores máximos y mínimos de crecimiento de cada planta.*

Esta clase, cuenta con métodos que harán las comprobaciones de cada subsistema, así como otras funciones auxiliares de lectura y una de escritura, que utilizará el HMI más adelante. También, cuenta con otras variables auxiliares públicas, que serán accesibles por todo el programa, al contrario que las privadas, que solo se tiene acceso mediante los métodos definidos (Figura 22).

```
/*
  Planta.h - librería de definición de una clase. Dicha clase se trata de
una planta,
  con las condiciones necesarias para su crecimiento. Los métodos que usa
están definidos
  en el código principal, usando en este caso funciones y variables
propias de este.
```

```
*/

//Se comprueba si ya estaba añadida esta librería para no añadirla dos
veces
#ifndef Planta_h
#define Planta_h

//Biblioteca general de Arduino
#include "Arduino.h"

class Planta {

  //Declaración de variables privadas, solo accesibles desde los métodos
públicos
  private:
    int ID;
    int LuzHoras;
    int TempMax;
    int TempMin;
    int HumMax;
    int HumMin;

  //Declaración de variables públicas, accesibles desde cualquier parte
del programa
  public:
    int AuxLuz;
    int AuxAgua;
    int AuxTemp;

    //Métodos públicos de la clase, accesibles desde cualquier parte del
programa
    Planta (int n); //Constructo, creación de objeto
    void SetPlanta (int h, int tmax, int tmin, int humax, int humin);
    void ContadorLuz (int n);
    void ComprobarAgua (int hum);
    void ComprobarTemp (int temp);
    int LeerLuz ();
    int LeerHumMax ();
    int LeerHumMin ();
    int LeerTempMin ();
    int LeerTempMax ();
};

#endif
```

Figura 22 *– Declaración de la clase "Planta" en una librería nueva de Arduino. Se declaran los métodos de manera pública, así como algunas variables auxiliares que pueden ser llamados desde cualquier parte del código principal. Inicialmente se declaran también, de manera privada, los datos asociados a cada objeto creado de esta clase.*

Ahora bien, cada método se ha definido en el programa principal, ya que es bastante dependiente del número de objetos declarados y de la funcionalidad que se le ha dado en este caso (Figura 23).

```
void Planta::ContadorLuz (int n) {
  if (n == 1) AuxLuz++; //Suma una vez cada 10 minutos de incidencia de
luz
  if (AuxLuz == 144) AuxLuz = 0; //Reinicia el contador cada 24h

  if (n == 1) { //En caso de haber luz
    LedOff (ID); //Se llama a la función que apaga los leds asociados a
esta planta
    //En caso de haber una cobertura para sombra se comprobaría si hay
exceso de luz
  }
  else if (AuxLuz < LuzHoras) { //En caso de necesitar más luz
    LedOn (ID); //Se llama a la función que enciende los leds asociados a
esta planta
  }
}

void Planta::ComprobarAgua (int hum) {
  AuxAgua = int (100 * (1 - (hum / 1023))); //Se añade la humedad actual
a la variable auxiliar en un formato de porcentaje

  if (AuxAgua < HumMin) RiegoOn (ID); //Enciende el riego de esta planta
  else if (AuxAgua > ((HumMax + HumMin)/2) && digitalRead (ID) != LOW)
RiegoOff (ID);
  //Se desea llegar a un valor óptimo, no entrar unicamente en el rango
}

void Planta::ComprobarTemp (int temp) {
  AuxTemp = temp; //Almacena el valor de temperatura actual

  //Comprueba si necesita ventilación mediante una media de temperatura
de ambas plantas
  if (temp > TempMax) { if (digitalRead (Fan2Pin == LOW)){ FanOn (ID);} }
  else if (temp < TempMedia && digitalRead (Fan1Pin) != LOW) FanOff (ID);
  //Comprueba si necesita calefacción mediante una media de temperatura
de ambas plantas
  if (temp < TempMin) { if (digitalRead (HeatPin == LOW)) HeatOn (ID); }
  else if (temp > TempMedia && digitalRead (HeatPin) != LOW) HeatOff
(ID);
}
```

Figura 23 *– Fragmento de código Arduino en el que se definen los métodos asociados a la clase "Planta". Estos métodos actualizan las variables auxiliares con las que se comprueba el estado de las plantas, además de ejecutar las medidas necesarias para mantener un estado óptimo para el crecimiento de los vegetales.*

Como se había indicado anteriormente, con la información obtenida por los sensores, el procesador va a encender o apagar los actuadores correspondientes para que el sistema mantenga unas condiciones óptimas. Para hacer esto se han creado funciones con una nomenclatura similar, un identificativo del actuador, seguido del estado al que se quiere llegar (Figura 24). Como ejemplo, se puede observar la función "LedOn", que encenderá el sistema de leds asociado al objeto desde el que se llama.

```
//-------Funciones auxiliares. Control de bomba-------
void RiegoOn (int n){
  if (n == Valv1Pin){
    if (digitalRead (BombaPin) == LOW){
      digitalWrite (Valv2Pin, HIGH);
      while (digitalRead (Valv2Pin) == LOW) {}
      digitalWrite (BombaPin, HIGH);

      lcdActionsUpdate ("ValvOn", 1);
      output_HMI.setCharAt (2, '1');
      action = true;
    }
    else {
      if (digitalRead (Valv1Pin == HIGH)) {
        digitalWrite (Valv1Pin, LOW);

        lcdActionsUpdate ("ValvOn", 1);
        output_HMI.setCharAt (2, '1');
        action = true;
      }
    }
  }
  else{
    if (digitalRead(BombaPin == LOW)) {
      digitalWrite (Valv1Pin, HIGH);
      while (digitalRead (Valv1Pin) == LOW) {}
      digitalWrite (BombaPin, HIGH);

      lcdActionsUpdate ("ValvOn", 1);
      output_HMI.setCharAt (2, '1');
      action = true;
    }
    else {
      if (digitalRead (Valv2Pin == HIGH)) {
        digitalWrite (Valv2Pin, LOW);

        lcdActionsUpdate ("ValvOn", 3);
        output_HMI.setCharAt (4, '1');
        action = true;
      }
    }
  }
}

void RiegoOff (int n){
  if (n == Valv1Pin){
    if (BombaPin == HIGH){
      if (digitalRead (Valv2Pin == LOW)) {
        digitalWrite (Valv1Pin, LOW);
```

```
       lcdActionsUpdate ("ValvOff", 1);
       output_HMI.setCharAt (2, '0');
       action = true;
      }
      else {
        digitalWrite (BombaPin, LOW);
        while (digitalRead (BombaPin) == HIGH) {}
        digitalWrite (Valv2Pin, LOW);
        digitalWrite (Valv1Pin, LOW);

        lcdActionsUpdate ("ValvOff", 1);
        output_HMI.setCharAt (2, '0');
        action = true;
      }
    }
  }
  else {
    if (BombaPin == HIGH){
      if (digitalRead (Valv2Pin == LOW)) {
        digitalWrite (Valv2Pin, LOW);

       lcdActionsUpdate ("ValvOff", 3);
       output_HMI.setCharAt (4, '0');
       action = true;
      }
      else {
        digitalWrite (BombaPin, LOW);
        while (digitalRead (BombaPin) == HIGH) {}
        digitalWrite (Valv2Pin, LOW);
        digitalWrite (Valv1Pin, LOW);

        lcdActionsUpdate ("ValvOff", 3);
        output_HMI.setCharAt (4, '0');
        action = true;
      }
    }
 }
}

//-------Funciones auxiliares. Control de ventilador-------
void FanOn (int n) {
  digitalWrite (Fan1Pin, HIGH); //Cambia el estado de un ventilador
  digitalWrite (Fan2Pin, HIGH);

  if (n == Valv1Pin) lcdActionsUpdate ("Tempv", 1);
  else lcdActionsUpdate ("Tempv", 2);
  output_HMI.setCharAt (6, '1');
  action = true;
```

```
}

void FanOff (int n) {
  digitalWrite (Fan1Pin, LOW);
  digitalWrite (Fan2Pin, LOW);

  if (n == Valv1Pin) lcdActionsUpdate ("Temp-", 1);
  else lcdActionsUpdate ("Temp-", 2);
  output_HMI.setCharAt (6, '0');
  action = true;
}


//-------Funciones auxiliares. Control de calefactor-------
void HeatOn (int n) {
  digitalWrite (HeatPin, HIGH); //Cambia el estado del calefactor

  if (n == Valv1Pin) lcdActionsUpdate ("Temp^", 1);
  else lcdActionsUpdate ("Temp^", 2);
  output_HMI.setCharAt (6, '2');
  action = true;
}

void HeatOff (int n) {
  digitalWrite (HeatPin, LOW);

  if (n == Valv1Pin) lcdActionsUpdate ("Temp-", 1);
  else lcdActionsUpdate ("Temp-", 2);
  output_HMI.setCharAt (6, '0');
  action = true;
}


//-------Funcion auxiliar. Control de luces-------
void LedOn (int n) {
  if (n == Valv1Pin) {
    if (digitalRead (Led1Pin == LOW)) {
      digitalWrite (Led1Pin, HIGH); //Cambia el estado de los leds de una
planta

      lcdActionsUpdate ("LEDOn", 1);
      output_HMI.setCharAt (8, '1');
      action = true;
    }
  }
  if (n == Valv2Pin) {
    if (digitalRead (Led1Pin == LOW)) {
          digitalWrite (Led2Pin, HIGH);
```

```
      lcdActionsUpdate ("LEDOn", 2);
      output_HMI.setCharAt (10, '1');
      action = true;
    }
  }
}

void LedOff (int n) {
  if (n == Valv1Pin) {
    if (Led1Pin == HIGH) {
      digitalWrite (Led1Pin, LOW);

      lcdActionsUpdate ("LEDOff", 1);
      output_HMI.setCharAt (8, '0');
      action = true;
    }
  }
  if (n == Valv2Pin) {
    if (Led2Pin == HIGH) {
      digitalWrite (Led2Pin, LOW);

      lcdActionsUpdate ("LEDOff", 2);
      output_HMI.setCharAt (10, '0');
      action = true;
    }
  }
}

void FanOff () {
  digitalWrite (Fan1Pin, LOW);
  digitalWrite (Fan2Pin, LOW);

  lcdActionsUpdate ("Temp-", 1);
  lcdActionsUpdate ("Temp-", 2);
  output_HMI.setCharAt (6, 0);
  action = true;
}


//-------Funciones auxiliares. Control de calefactor-------
void HeatOn () {
  digitalWrite (HeatPin, HIGH); //Cambia el estado del calefactor

  lcdActionsUpdate ("Temp^", 1);
  lcdActionsUpdate ("Temp^", 2);
  output_HMI.setCharAt (6, 2);
  action = true;
}
```

```
void Hentoff () {
  digitalWrite (HeatPin, LOW);

  lcdActionsUpdate ("Temp-", 1);
  lcdActionsUpdate ("Temp-", 2);
  output_HMI.setCharAt (6, 0);
  action = true;
}


//-------Función auxiliar. Control de luces-------
void LedOn (int n) {
  if (n == Valv1Pin) {
    if (digitalRead (Led1Pin == LOW)) {
      digitalWrite (Led1Pin, HIGH); //Cambia el estado de los leds de una
planta

      lcdActionsUpdate ("LEDOn", 1);
      output_HMI.setCharAt (8, 1);
      action = true;
    }
  }
  if (n == Valv2Pin) {
    if (digitalRead (Led1Pin == LOW)) {
          digitalWrite (Led2Pin, HIGH);

      lcdActionsUpdate ("LEDOn", 2);
      output_HMI.setCharAt (8, 1);
      action = true;
    }
  }
}

void LedOff (int n) {
  if (n == Valv1Pin) {
    if (Led1Pin == HIGH) {
      digitalWrite (Led1Pin, LOW);

      lcdActionsUpdate ("LEDOff", 1);
      output_HMI.setCharAt (10, 0);
      action = true;
    }
  }
  if (n == Valv2Pin) {
    if (Led2Pin == HIGH) {
      digitalWrite (Led2Pin, LOW);

      lcdActionsUpdate ("LEDOff", 2);
      output_HMI.setCharAt (10, 0);
```

```
      action = true;
    }
  }
}
```

Figura 24 *- Fragmento de código Arduino en el que se definen las diferentes funciones asociadas a cada actuador. En este caso existen dos funciones para cada componente, para apagar y para encender. Se utilizan el número de pin en el que están conectados los componentes, almacenados en sus variables correspondientes en la configuración inicial del programa. También modifica una cadena declarada globalmente para mostrar el cambio realizado mediante el LCD, así como actualiza el estado de cada componente para la comunicación con el HMI.*

Todas las salidas están conectadas a un pin digital, con una tensión de 5 V y una corriente no superior a 20 mA. Es debido a esto que, estas salidas activarán los circuitos correspondientes de cada actuador, que se mostrarán más adelante.

En cada una de estas funciones se cambia el valor de un booleano, para indicar que el sistema ha hecho un cambio. Esto se usará para la actualización de un LCD, que, en caso de haber cambios, mostrará que actuador se ha activado o desactivado en cada una de las plantas. Para representar esta información en el LCD, se añaden fragmentos de una cadena de caracteres a un string asociado a una de las líneas de la pantalla. Los strings son también cadenas de caracteres, que se actualizarán continuamente para este propósito, y también se les dará uso más adelante en la comunicación serial con la interfaz gráfica.

En caso de no haber cambios por parte de los actuadores, esta pantalla se va a utilizar para mostrar el estado de las plantas (Figura 25). En cada ciclo del bucle principal se va a alternar entre el primer vegetal y el segundo, para mostrar sus valores exactos de humedad, horas de luz y temperatura. Cabe mencionar que esto es una representación secundaria, puesto que el HMI va a ser donde se vean estos datos como método principal.

```
 //Actualizacion del LCD
  if (action == false) lcdDataUpdate (); // Actualización de datos de las
plantas
  lcdUsage (); //Escritura de las strings actual en el lcd
```

Figura 25 *– Fragmento de código Arduino en el que se comprueba mediante una variable auxiliar si se debe mostrar por el LCD el estado de las plantas o las actualizaciones de los estados de los actuadores en la última iteración de bucle principal.*

En la función “lcdUsage ()” se escriben los strings actuales en el LCD y se borran para reiniciar su escritura en un nuevo ciclo del bucle principal. Generalmente, en los ciclos que no hay cambios en los actuadores, la función “lcdDataUpdate” actualiza las strings del LCD para mostrar el estado de una de las plantas (Figura 26). En caso contrario, las strings mantienen la forma que se les ha dado en el momento de cambiar los estados digitales de las salidas.

```
void lcdUsage () {
  lcd.clear ();

  //Imprime las cadenas de carcteres en el LCD una por una
  lcd.setCursor (0, 0);
  lcd.print (lcdRow1);
  lcd.setCursor (0, 1);
  lcd.print (lcdRow2);
  lcd.setCursor (0, 2);
  lcd.print (lcdRow3);
  lcd.setCursor (0, 3);
  lcd.print (lcdRow4);
```

```
  //Reinicia las cadenas de caracteres
  lcdRow1 = "";
  lcdRow2 = "";
  lcdRow3 = "";
  lcdRow4 = "";
  action = false; //Reinicia variable auxiliar de cambio de estado en los
actuadores
}
```

Figura 26 *– Fragmento del código principal del código de Arduino en el que se define la función encargada de mostrar las cadenas actualizadas a lo largo de cada ciclo en el LCD. También reinicia estas cadenas a un estado base para la siguiente iteración de ciclo en el que se actualice nuevamente.*

Por último, el programa debe mantener una comunicación serial con la interfaz gráfica. Esta comunicación se basa en el envío de una trama de bits, donde el conjunto de la trama tiene un significado codificado de la manera que se configure. En este caso, se pretende usar la trama para enviar un mensaje con la información deseada separada por caracteres alfabéticos y de puntuación.

En el envío de la trama desde el Arduino al HMI, se pretende usar el siguiente formato:

$$\#H0A0B0C0D0x,x,x,x,x$$

En esta cadena se envían dos tipos de valores, los estados de los actuadores, y las lecturas proporcionadas por los sensores. Ambos tipos de datos deberán ser enviados en cada ciclo del bucle principal, para actualizar continuamente el funcionamiento del sistema.

En primer lugar, se encuentra el identificativo del inicio de la trama, "#H". Seguido de esto se encuentran en orden los estados de cada válvula, el estado de los actuadores correspondientes al sistema de temperatura y los estados de cada agrupación de leds. Estos datos están separados por un carácter alfabético, para hacer más sencilla la identificación por parte del código en el HMI.

En segundo lugar, se encuentran los valores proporcionados por cada uno de los sensores correspondientes a las variables a controlar. En este caso, los valores están separados por comas, para ser separados y representados directamente en la interfaz gráfica. Esta separación permite también que el programa pueda separar los valores independientemente de si son de uno o dos dígitos. Toda esta lectura se verá más adelante en el código del HMI.

La comunicación tiene que ser bidireccional, sin embargo, desde la aplicación gráfica solo se va a enviar información cuando se pretendan cambiar los valores límites de desarrollo de una de las plantas. Debido a esto, al final de cada ciclo del bucle principal, el Arduino comprobará si hay alguna trama recibida para leer mediante la función "SerialEvent ()" (Figura 27).

```
void serialEvent () {
  //Función llamada al final de cada iteración del loop
  while (Serial.available ()) { //Funciona en caso de haber un mensaje
entrante
    char inputChar = Serial.read (); //Variable auxiliar que lee cada
byte
    input_HMI += inputChar; //Se añade cada byte leído en forma de
caracteres en una cadena global

    if (inputChar == '\n') inputDone = true; //Se indica que se ha
finalizado la lectura
```

```
  }
}
```

***Figura 27** – Fragmento de código Arduino en el que se comprueba una recepción de datos por parte del HMI. En caso de haber datos por leer, el programa almacena byte a byte en forma de caracteres en una cadena global del programa.*

Esta función es parecida a una interrupción, puede funcionar sin ser llamada por el bucle principal. Sin embargo, solo se ejecuta al final de cada iteración de ciclo. En este punto, si hay una trama de información que ha llegado al Arduino, la función separará byte a byte para convertir la agrupación de unos y ceros en un string formado por caracteres.

Al igual que con el envío, en la recepción también hay un formato específico que seguirán Arduino y HMI. Este formato es el siguiente:

$\#Ax,x,x,x,x$

Donde el primer carácter es identificativo del inicio de trama y el segundo define a que planta se refieren los datos siguientes. Al haber dos plantas, puede haber una "A" o una "B", siendo el primer objeto de la clase o el segundo.

El resto de los valores están separados por una coma, para así identificar la separación, independientemente de si hay uno o dos dígitos (Figura 28).

```
void serialConfig () {
  //Configuracion de los valores de crecimiento de cada planta desde el
HMI
  String n[6]; //Se crea una matriz de strings auxiliares
  int i = 0; //Variable auxiliar

  if (input_HMI.indexOf ("#A") != -1) { //Correspondiente a la planta 1
    input_HMI = input_HMI.substring (2); //Se elimina el identificativo
    //Se separan de la cadena cada uno de los valores separados por comas
    while (input_HMI.indexOf (",") != -1) {
      i++;
      n[0] = input_HMI.substring (0, input_HMI.indexOf (","));
      input_HMI = input_HMI.substring (input_HMI.indexOf (",") + 1);
      n[i] = n[0].toInt ();
    }
    n[5] = input_HMI.toInt ();
    //Se configuran los nuevos valores de la planta 1
    Planta1.SetPlanta (n[1].toInt (), n[2].toInt (), n[3].toInt (),
n[4].toInt (), n[5].toInt ());
  }
  else if (input_HMI.equals("#B") !=1) { //Mismo proceos anterior pero en
la planta 2
    input_HMI = input_HMI.substring (2);
    while (input_HMI.indexOf (",") != -1) {
      i++;
      n[0] = input_HMI.substring (0, input_HMI.indexOf (","));
      input_HMI = input_HMI.substring (input_HMI.indexOf (",") + 1);
      n[i] = n[0].toInt ();
    }
    n[5] = input_HMI.toInt ();
```

```
    Planta2.SetPlanta (n[1].toInt (), n[2].toInt (), n[3].toInt (),
n[4].toInt (), n[5].toInt ());
  }
}
```

***Figura 28** – Fragmento del código Arduino en el que se separa y almacena los valores recibidos de parte del HMI. Estos valores son los correspondientes a las variables de desarrollo de los objetos "Planta", por lo que utiliza el método adecuado ".SetPlanta ()" para la actualización.*

En la función "SerialConfig ()", se separan los valores mediante strings y funciones asociadas con este tipo de variable, para posteriormente asignar estos valores a los objetos mediante los métodos "Planta1.SetPlanta ()" y "Planta2.SetPlanta ()".

La función ".IndexOf (" ")" devuelve el valor de la primera posición en la que se encuentre una agrupación de caracteres concreta en la cadena especificada, o un -1 en caso de encontrarla. Por otro lado, la función ".substring ( )" devuelve una cadena cortada desde la posición indicada, hasta el final de la cadena o hasta cumplir con una longitud dada. Estas funciones se utilizarán de la misma manera en el código correspondiente al HMI.

6.2. HMI

En la interfaz gráfica se desea que se muestre el estado del sistema, por lo que deberá representar los datos mostrados y el estado de los actuadores. Para la representación, se van a usar figuras que representan los elementos que se encuentran dentro del invernadero, y un recuadro para indicar los valores obtenidos desde el Arduino (Figura 29).

Además de esta información, el HMI tiene una serie de botones, cono los que se puede interactuar con el sistema. De aquí se destacan los botones de configuración, usados para enviar la trama de información deseada con unos cuadros de texto para rellenar con el valor deseado.

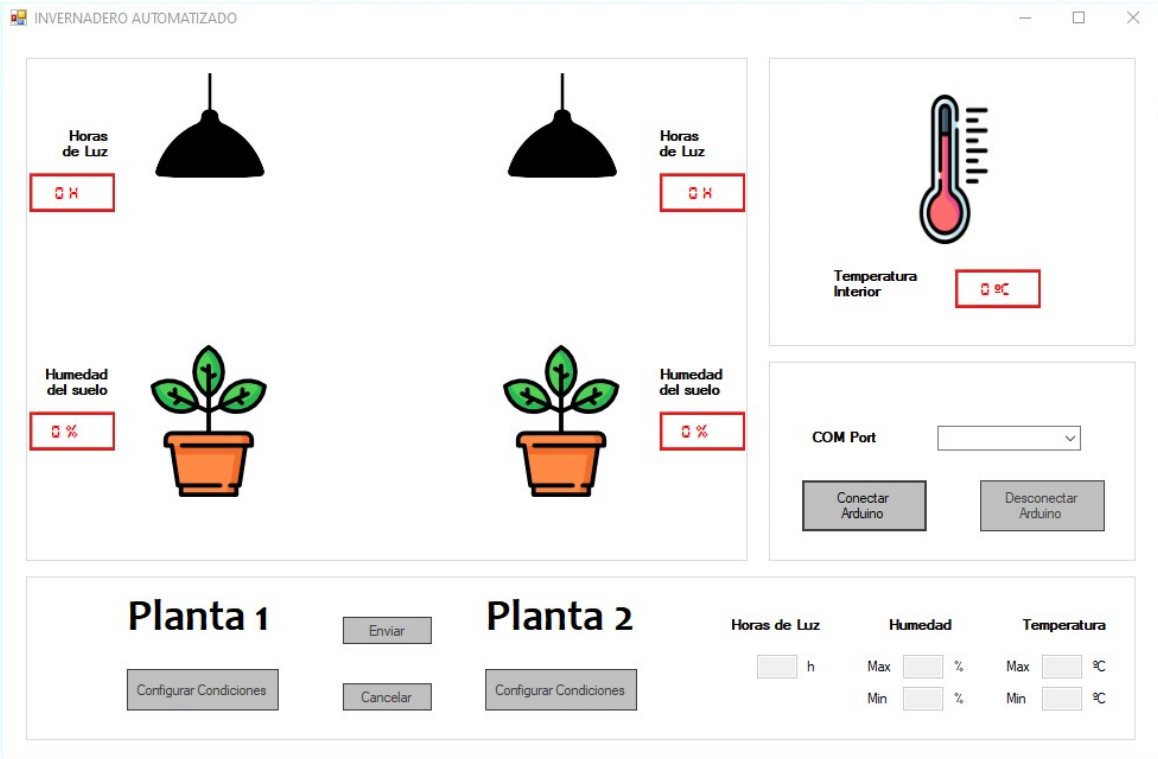


Figura 29 *– Resultado final del HMI. Se muestra el caso inicial, donde los actuadores están apagados y la conexión con el Arduino no se ha ejecutado todavía. Los valores de cada sensor están a 0 y cambiarán tan pronto como empiece a llegar información desde el Arduino.*

Esta aplicación se basa en la comunicación, por lo que cuenta también con un cuadro de mando para la conexión correcta con el Arduino.

La información recibida en la interfaz deberá ser separada teniendo en cuenta el formato de envío explicado anteriormente (Figura 30).

```
private void serialPort1_DataReceived(object sender,
SerialDataReceivedEventArgs e)
{ //Función ejecutada mediante el evento de recepción de información
    while (serialPort1.IsOpen && serialPort1.BytesToRead > 0)
    {//Mientras haya datos para leer y el puerto de comunicación esté
abierto
        string outputNANO = serialPort1.ReadLine(); //Almacenamiento
de la trama recibida
        outputNANO = outputNANO.Trim(); //Se eliminan los espacios de
la cadena

        if (outputNANO.IndexOf("#H") != -1) //Identificación del
inicio de la trama correcta
        {
            Invoke(new MethodInvoker(() => { //Asegura que se puedan
cambiar datos de otros objetos

                int n;
                if (outputNANO.IndexOf("#H0") != -1) //Caso de primera
válvula apagada
```

```
                {//Se vuelven invisibles los objetos del HMI
correspondientes
                    pictureBoxValv1On.Visible = false;
                    pictureBoxRiego1.Visible = false;
                    if (pictureBoxValv1On.Visible == false &&
pictureBoxValv2On.Visible == false)
                    {
                        pictureBoxBombaOn.Visible = false;
                    }
                }
                else if (outputNANO.IndexOf("#H1") != -1) //Caso de
primera válvula encendida
                {//Se vuelven visibles los objetos correspondientes
del HMI
                    pictureBoxValv1On.Visible = true;
                    pictureBoxRiego1.Visible = true;
                    pictureBoxBombaOn.Visible = true;
                }
                if (outputNANO.IndexOf("A0") != -1) //Caso de segunda
válvula apagada
                {
                    pictureBoxValv2On.Visible = false;
                    pictureBoxRiego2.Visible = false;
                    if (pictureBoxValv1On.Visible == false &&
pictureBoxValv2On.Visible == false)
                    {
                        pictureBoxBombaOn.Visible = false;
                    }
                }
                else if (outputNANO.IndexOf("A1") != -1) //Caso de
segunda válvula encendida
                {
                    pictureBoxValv2On.Visible = true;
                    pictureBoxRiego2.Visible = true;
                    pictureBoxBombaOn.Visible = true;
                }
                if (outputNANO.IndexOf("B0") != -1) //Caso de
temperatura estable
                {
                    pictureBoxFlechaUp.Visible = false;
                    pictureBoxFlechaDown.Visible = false;
                    pictureBoxFanOn.Visible = false;
                    pictureBoxHeatOn.Visible = false;
                }
                else if (outputNANO.IndexOf("B1") != -1) //Caso de
ventilación activada
                {
                    pictureBoxFlechaDown.Visible = true;
                    pictureBoxFanOn.Visible = true;
```

```
                    pictureBoxFlechaUp.Visible = false;
                    pictureBoxHeatOn.Visible = false;
                }
                else if (outputNANO.IndexOf("B2") != -1) //Caso de
calefacción encendida
                {
                    pictureBoxFlechaUp.Visible = true;
                    pictureBoxHeatOn.Visible = true;
                    pictureBoxFlechaDown.Visible = false;
                    pictureBoxFanOn.Visible = false;
                }
                if (outputNANO.IndexOf("C0") != -1) //Caso de primer
conjunto de leds apagados
                {
                    pictureBoxLed1.Visible = false;
                }
                else if (outputNANO.IndexOf("C1") != -1) //Caso de
primer conjunto de leds encendidos
                {
                    pictureBoxLed1.Visible = true;
                }
                if (outputNANO.IndexOf("D0") != -1) //Caso de segundo
conjunto de leds apagados
                {
                    pictureBoxLed2.Visible = false;
                }
                else if (outputNANO.IndexOf("D1") != -1) //Caso de
segundo conjunto de leds encendidos
                {
                    pictureBoxLed2.Visible = true;
                }

                //Se seccionan los valores de la cadena
                n = outputNANO.IndexOf(",");
                string outputC = outputNANO.Substring(11, n - 11);
                labelTempValue.Text = (outputC + " °C"); //Se cambia
el valor de la temperatura actual en el HMI

                outputNANO = outputNANO.Substring(n + 1);
                n = outputNANO.IndexOf(",");
                outputC = outputNANO.Substring(0, n);
                labelHum1Value.Text = (outputC + " %"); //Se cambia el
valor de la humedad actual de la primera planta en el HMI

                outputNANO = outputNANO.Substring(n + 1);
                n = outputNANO.IndexOf(",");
                outputC = outputNANO.Substring(0, n);
                labelHum2Value.Text = (outputC + " %"); //Se cambia el
valor de la humedad actual de la segunda planta en el HMI
```

```
                outputNANO = outputNANO.Substring(n + 1);
                n = outputNANO.IndexOf(",");
                outputC = outputNANO.Substring(0, n);
                labelL1Value.Text = ((float.Parse(outputC) / 10) + "
h"); //Se cambia el valor de las horas de luz actuales de la segunda
planta en el HMI

                outputNANO = outputNANO.Substring(n + 1);
                labelL2Value.Text = ((float.Parse(outputNANO) / 10) +
" h"); //Se cambia el valor de las horas de luz actuales de la segunda
planta en el HMI
            }));

        }
    }
}
```

Figura 30 *– Fragmento del código del HMI en lenguaje C#. Se trata de la recepción de trama por parte de la comunicación serial, así como la lectura de esta trama en formato de caracteres. En primer lugar, lee los valores asignados a cada posición, teniendo en cuenta el formato escogido desde el Arduino. En esta etapa también modifica la visibilidad de los objetos dentro de la interfaz gráfica según los valores que ha recibido. En segundo lugar, separa y almacena los valores de mediciones de los sensores en los objetos correspondientes dentro de la interfaz gráfica.*

En este caso, cada estado de cada actuador tiene como consecuencia la activación o desactivación de las imágenes que las representan. En cada caso, se tiene en cuenta la lógica del sistema, para no mostrar un caso imposible.

Los valores de los sensores son separados por comas para poder cortar la parte de la cadena deseada y representarla junto a su unidad en los cuadros de valores de la aplicación gráfica. Las funciones que se usa en esta parte del programa son las mismas que se usa en Arduino para hacer la misma tarea.

Para asegurar la integridad del programa, se deshabilitan todos los botones al inicio de la aplicación, a excepción de “Conectar Arduino”. Son habilitados únicamente cuando la conexión al procesador se ha realizado con éxito.

6.3. Componentes electrónicos

Como se ha comentado anteriormente, los actuadores del sistema tendrán un circuito con alimentación externa al Arduino, pero controlados por la salida digital de este. Para que esto sea posible se va a diseñar el circuito con transistores BJT, para que trabajen conmutando entre los estados de corte y saturación.

El esquema base para esto consta de la salida digital de Arduino conectada la base de un transistor NPN. El emisor del transistor estará conectado a la referencia del circuito, mientras que en la base estará el actuador y la tensión externa, que proporcionará la corriente necesaria para el correcto funcionamiento del componente (Figura 31).

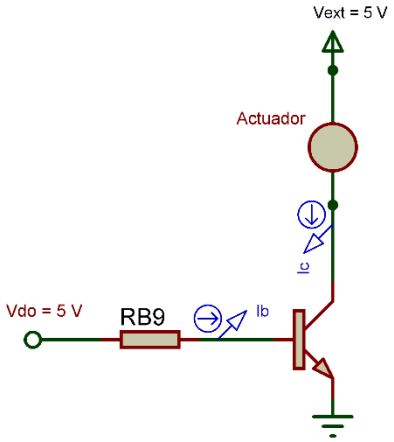


Figura 31 *– Esquema ejemplo de conexión de un actuador de mayor potencia controlado por un transistor NPN conectado a una salida digital del Arduino. La rama superior es el colector, donde se conecta la carga. La rama izquierda es la base, con una corriente controlada mediante la resistencia de base. El emisor está conectado directamente a referencia.*

En este circuito, se desea que el transistor entre en saturación, y para logarlo hay que calcular el valor de resistencia necesaria en la base del transistor para que entre en saturación.

A excepción de la placa calefactora de cerámica, los componentes por separado no superan una corriente de 300 mA. Debido a esto, se va a utilizar un BD137, un transistor capaz de soportar en el colector una corriente máxima de 1.5 A y 100 V de tensión colector-emisor. La corriente máxima que soporta el colector es justo la misma que necesita el MCH, la placa calefactora, por lo que se buscará otro componente para este actuador.

En el caso del BD137, según la hoja de características la ganancia (h_{fe}) y la tensión de de base-emisor (V_{be}) son de 40 y 1 V respectivamente [11]. Con estos datos se va a calcular la resistencia de base (R_b) conectada a la salida del Arduino (V_{do}), mediante la siguiente fórmula:

$$R_b = \frac{V_{do} - V_{be}}{I_b}$$

Donde la corriente de base (I_b), depende del consumo de cada componente conectado al colector del transistor (I_c) y de la ganancia mencionada anteriormente:

$$I_b = \frac{I_c}{h_{fe}}$$

Así pues, se calculan las resistencias de base para el circuito de actuadores, suponiendo la salida del Arduino con un valor de 5 V y sustituyendo en las ecuaciones mencionadas los valores de cada componente (Tabla 5).

Tabla 5 – Tabla con los resultados de corriente de base y resistencia de base necesarios para una corriente en el colector suficiente para cada uno de los actuadores.

Actuador	I_c (mA)	V_{do} (V)	I_b (mA)	R_b (kΩ)
Bomba de agua	300	5	7,5	0,53
Electroválvula	220	5	5,5	0,73
Ventilador	60	5	1,5	2,67

Con los resultados obtenidos se asegura que la corriente en el colector sea la correcta, utilizando el funcionamiento del transistor en saturación. Ninguna de las corrientes es superior a 20 mA, por lo que el Arduino debería poder proporcionar esa corriente en la base del transistor. Ahora bien, estos valores de resistencias son los teóricos, por lo que se escogen valores de resistencias normalizados de la serie E-12 inmediatamente inferiores. De esta manera, se recalculan las corrientes de base de cada actuador con las nuevas resistencias normalizadas, utilizando las fórmulas anteriores (Tabla 6).

Tabla 6 – Tabla con los resultados de corrientes de base obtenidos para las resistencias normalizadas. Estas corrientes aseguran el estado de saturación en el BJT.

Actuador	V_{do} (V)	$R_{b\text{-}norm}$ (kΩ)	I_b (mA)
Bomba de agua	5	0,47	10,64
Electroválvula	5	0,68	7.35
Ventilador	5	2,2	2,27

Con las nuevas corrientes de base se asegura un estado de saturación, por lo que las corrientes en el colector serán las que necesiten los actuadores.

De entre estos componentes, las electroválvulas, la bomba de agua y el ventilador se basan en aplicaciones de campos magnéticos. Para prevenir una descarga de la corriente inducida en el circuito, dañando los transistores e incluso el Arduino, se va a instalar un diodo flyback en paralelo (Figura 32). Esta configuración utiliza un diodo con una tensión de rotura y corriente directa máxima superiores a la que consume el componente con campo magnético.

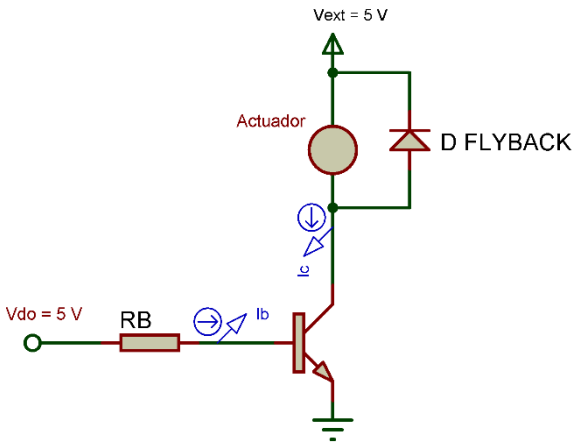


Figura 32 – Esquema de conexión de un transistor para el control de una carga en el colector mediante una salida controlada en la base. En este caso, la carga tiene conectada un diodo en paralelo en sentido contrario para evitar que las posibles corrientes inducidas vayan hacia el resto de los componentes.

Para la elección de los diodos se va a utilizar la corriente máxima de estos actuadores, en este caso la de la bomba de agua, con un valor de 300 mA. A esta corriente se la va a aplicar un fator de seguridad de 10, por lo que la corriente directa máxima que pueda soportar el diodo será de 3 A. Así pues, el modelo de diodo escogido es el 1N5408, capaz de soportar un máximo de 3 A en directa y hasta 1000 V hasta la ruptura [12].

En el caso de la placa calefactora MCH, la corriente que necesita es de un valor de una magnitud bastante mayor a la que puede proporcionar el Arduino en su salida. La ganancia necesaria por el transistor de control es inferior y su capacidad de corriente máxima en el colector es la misma que necesita el actuador. Por este motivo, el circuito para este caso será algo diferente.

Para el control del MCH se va a utilizar un relé a modo de interruptor mecánico para el circuito. El relé estará controlado a su vez por los mismos transistores que el resto de los actuadores, el BD137 (Figura 33).

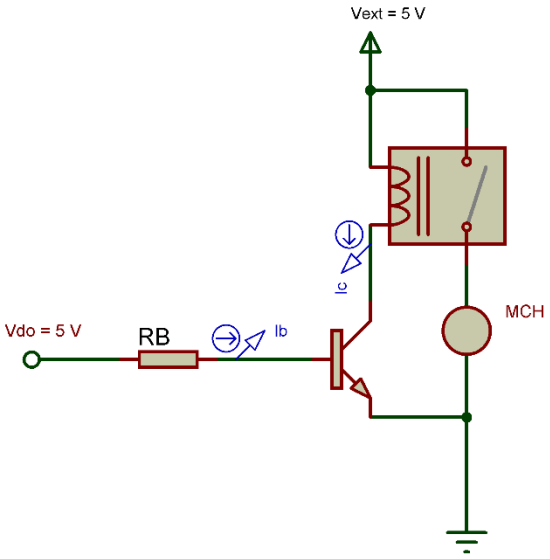


Figura 33 *– Circuito ejemplo de control de una placa calefactora MCH mediante un relé. El relé a su vez está controlado mediante un transistor con la base conectada a una salida de Arduino.*

Se va a utilizar un relé de la marca Finder, el modelo 36.11.9.005.4011. Este componente puede funcionar con una corriente continua, variando la impedancia equivalente de la bobina en función de la tensión aplicada. Según la hoja de características [13], para una tensión de 5 V que es con la que trabaja todo el sistema (V_{ext}), la impedancia equivalente es de 70 Ω ($R_{relé}$). En primer lugar, se calcula la corriente que necesita el relé (I_c) mediante la ley de Ohm:

$$I_c = \frac{V_{ext}}{R_{relé}}$$

$$I_c = \frac{5\ V}{70\ \Omega} = 71.43\ mA$$

Con esta corriente se calcula la corriente mínima que necesita el BJT en base (I_b) para saturar, teniendo una ganancia de 40 (h_{fe}):

$$I_b = \frac{I_c}{h_{fe}}$$

$$I_b = \frac{71.43\ mA}{40} = 1.79\ mA$$

Para esta corriente en la base de transistor hace falta una resistencia (R_b), teniendo en cuenta la tensión de salida del Arduino (V_{do}) y la tensión base-emisor del BJT (V_{be}):

$$R_b = \frac{V_{do} - V_{be}}{I_b}$$

$$R_b = \frac{5\ V - 1\ V}{1.79\ mA} = 2.23\ k\Omega$$

Con el valor de resistencia obtenido se busca el valor normalizado inmediatamente inferior y se comprueba el valor de corriente real teórico que habrá en la base del emisor:

$$I_b = \frac{V_{do} - V_{be}}{R_{b-norm}}$$

$$I_b = \frac{5\ V - 1\ V}{2.2\ \Omega} = 1.82\ mA$$

De esta manera, se asegura el funcionamiento del relé, que a su vez conmutará el circuito de la placa calefactora. Para mayor seguridad, se le va a añadir el mismo diodo que al esto de actuadores, en paralelo al relé para controlar las posibles corrientes inducidas.

Además de estas salidas, se cuenta también con las agrupaciones de leds. Para una salida de 5 V del Arduino, se había calculado anteriormente que se podían conectar dos leds de color rojo en cada salida. Por lo que se calculará el valor de una resistencia reguladora que permita el funcionamiento correcto de estos leds. Así pues, la resistencia (R_{Led}) que fije la corriente en la salida para una tensión de salida (V_{do}):

$$R_{Led} = \frac{V_{do} - V_{Led1} - V_{Led2}}{I_{Led}}$$

Donde los valores de tensión de cada led (V_{Led}) y su corriente (I_{Led}), son los mencionados anteriormente. De este modo:

$$R_{Led} = \frac{5\ V - 2.4\ V - 2.4\ V}{20\ mA} = 10\ \Omega$$

El valor obtenido es equivalente al de una resistencia normalizada, por lo que no hay cambiarlo en este caso.

Este circuito se repite una vez más en el sistema, puesto que son dos agrupaciones de leds independientes, una para cada planta. Con todo esto, la conexión de todos los componentes de salida al Arduino está diseñada (Figura 34).

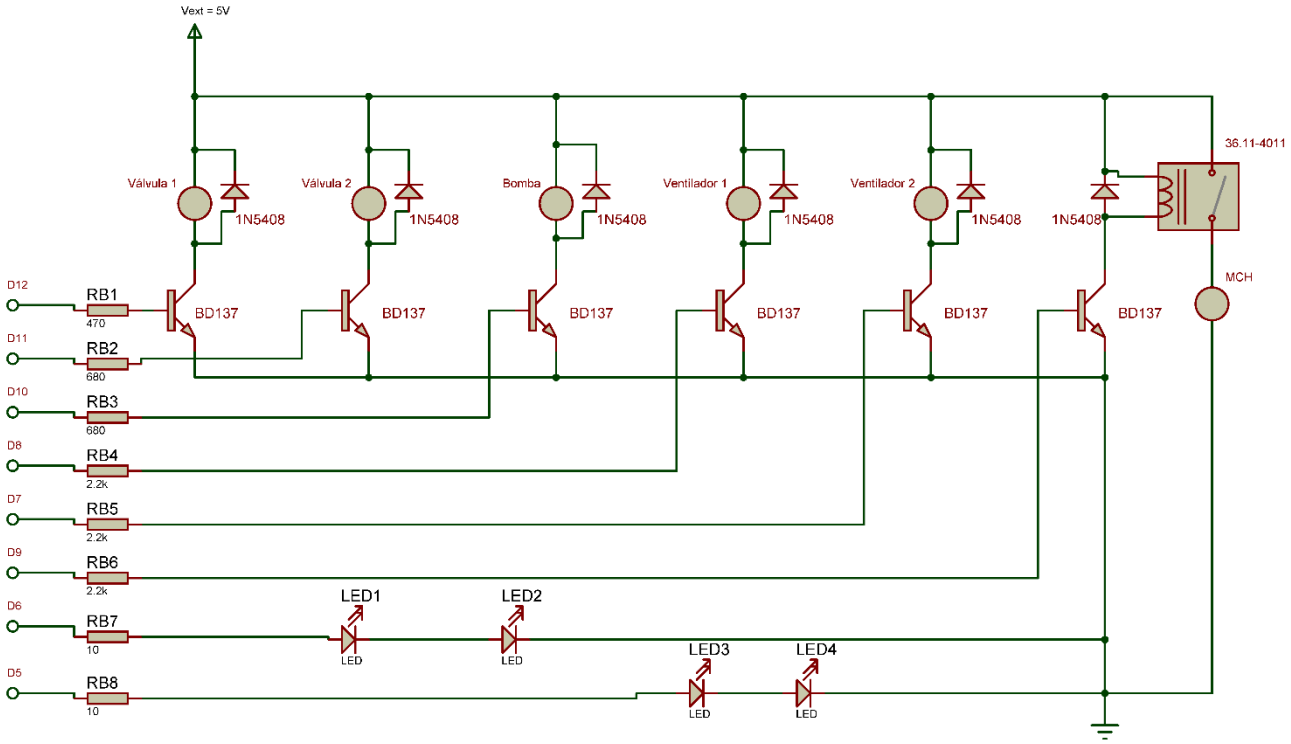


Figura 34 *– Circuito completo de las salidas del Arduino. Se pueden observar el valor de las resistencias de cada rama, asociadas a cada uno de los actuadores del sistema. También se indican el modelo de los componentes como transistores y relé.*

En el código se ha implementado el uso de un LCD no mencionado anteriormente. Este LCD es un display de 4 líneas y 20 columnas, con un módulo I2C integrado, que permite una conexión más sencilla al Arduino [14]. La conexión se hace mediante los pines de comunicación I2C, que son los pines SDA y SCL, que, a su vez, son los pines A5 y A4 (Figura 35).

Por último, todos los sensores tienen ya un módulo de conexión, ya sea mediante el LM393 o mediante un circuito con resistencia propia como es el DHT11. Por este motivo, la conexión a Arduino es directa (Figura 35).

Todos estos componentes suponen un consumo, por lo que más adelante se verificará la potencia consumida por todo el sistema. Además, se comprobará que tanto la fuente externa como el propio Arduino pueden proporcionar la corriente necesaria.

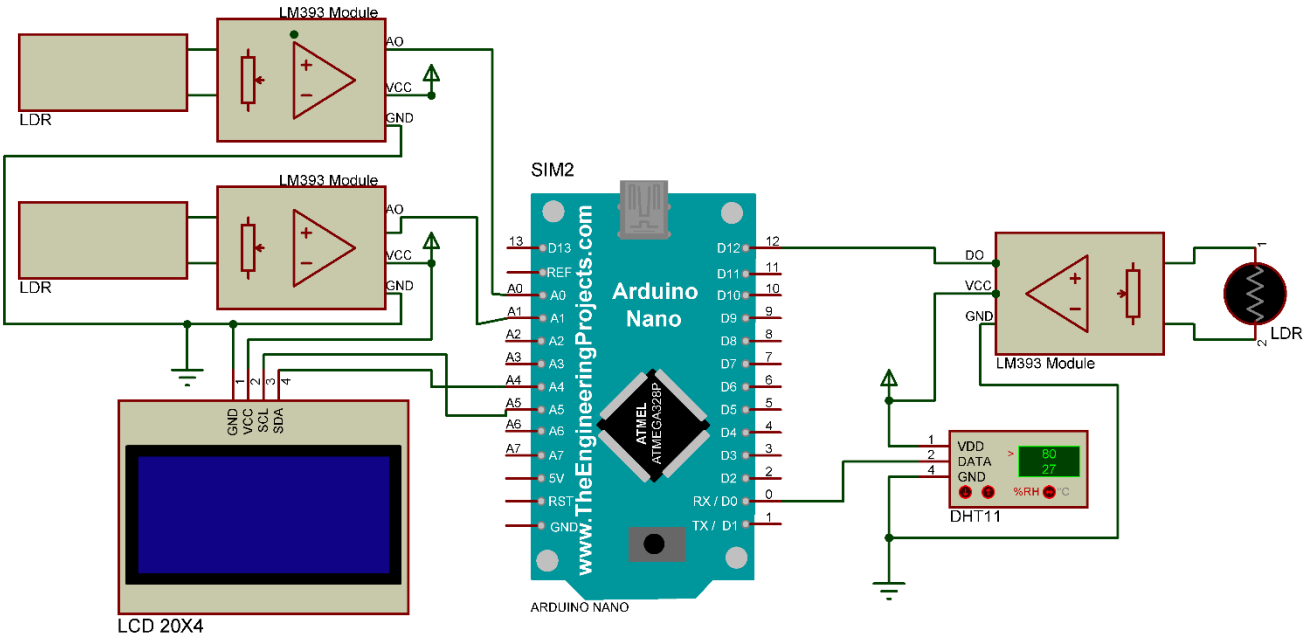


Figura 35 *– Esquema eléctrico de conexión de algunos componentes al Arduino NANO. Los dos sensores FC-28, sensores de humedad, están conectadas mediante un módulo comparativo basado en el amplificador operacional LM393 y una resistencia regulable a dos pines analógicos del procesador. El fotorresistor GL5593 usa el mismo módulo comparativo basado en LM393, con una salida digital. El DHT11, sensor de temperatura y humedad del aire está conectado directamente a un pin digital. Por último, el LCD está conectado mediante dos pines digitales gracias al módulo I2C que tiene incorporado.*

6.4. Medidas de la maqueta

Las plantas seleccionadas, el orégano y el cilantro, son plantas que en una maceta necesitan de un máximo de 30 cm de diámetro. Este tamaño es el que le corresponde a un solo tallo. Además, ambas plantas crecen hasta un máximo de 60 cm de alto.

A partir de estas dimensiones de las plantas se ha dimensionado la maqueta de invernadero. Suponiendo estas macetas como un cubo de lado de 30 cm, la maqueta debe dejar espacio para el resto de equipo de necesario (Figura 36).

Los sensores DHT11 y el fotorresistor irán apoyados en la viga horizontal central. Esta viga proporciona más estabilidad a la estructura, además de permitir un punto de apoyo para los componentes. De esta viga irán colgadas también las tuberías con las que se regará los cultivos.

En la parte exterior se encuentran bajo una cubierta el Arduino, así como una placa de montaje, o protoboard, donde irán montados los componentes electrónicos básicos como las resistencias y transistores. La bomba de agua estará bajo la cubierta exterior también, con acceso a un tanque de agua exterior mediante las tuberías.

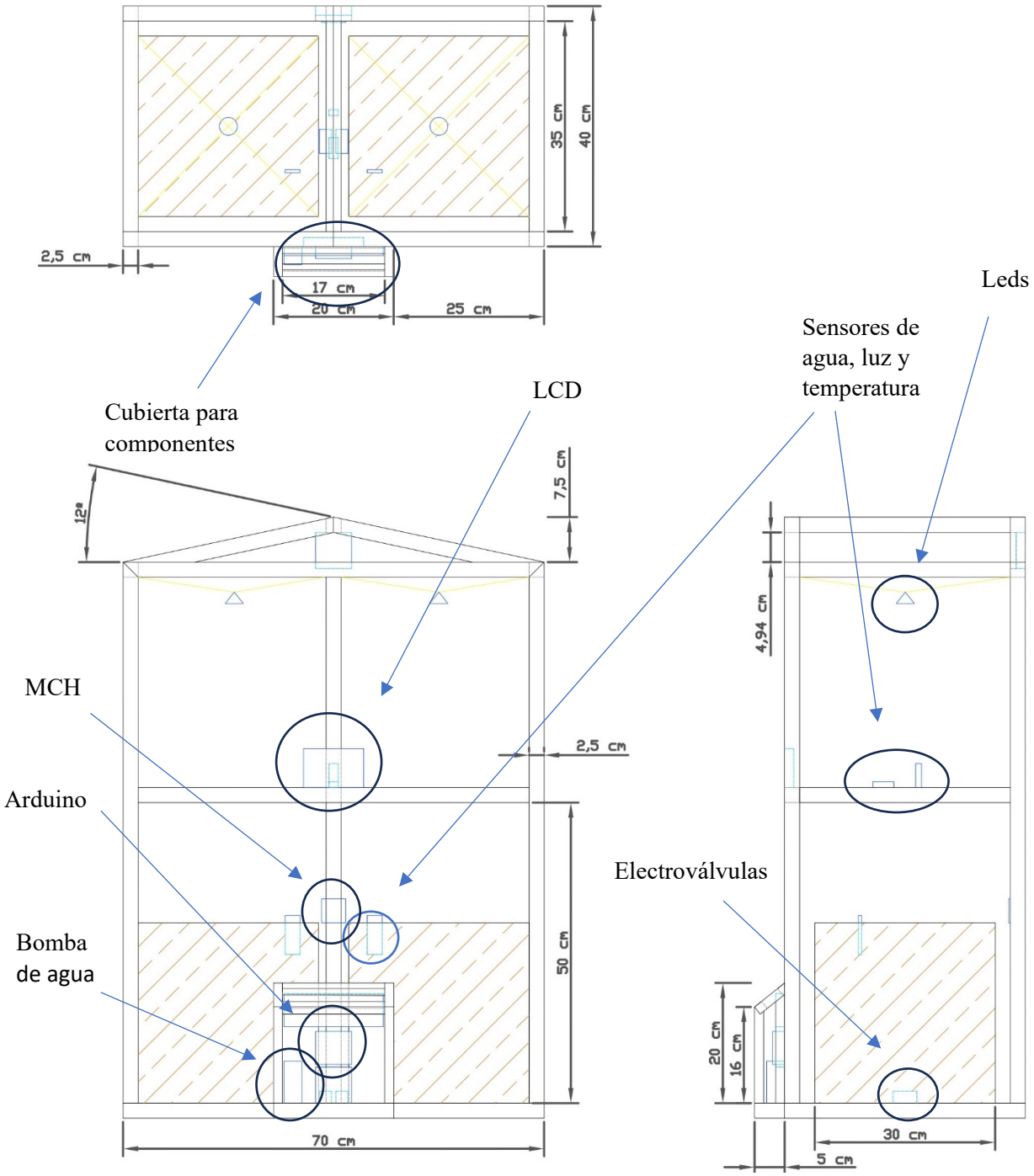


Figura 36 *– Planta alzado y perfil del plano final de la maqueta. Se indican sus dimensiones totales, el grosor de las vigas y se señalan los diferentes componentes del sistema. Las porciones de tierra donde se cultiva están marcadas con un relleno marrón, mientras que los componentes están dibujados con un trazo azul. Las lámparas de leds están colgadas mediante cuerdas, representadas con un trazo amarillo.*

En cuanto a los ventiladores, el de entrada se encuentra justo debajo del Arduino, mientras que el de salida se encuentra en la zona superior, incrustado entre de las vigas diagonales que forman el tejado y la vertical del lado contrario del Arduino. Las válvulas están entre las macetas, conectadas a la bomba de agua mediante tuberías. Por último, la placa calefactora se encuentra apoyada en la viga central, justo por encima de las macetas.

El resto de las componentes estarán entre las macetas o apoyados en las vigas verticales, como es el caso de la placa térmica de cerámica.

Todas las vigas principales, tanto horizontales como verticales, serán de sección cuadrada, con un grosor de 2.5 cm de lado. La cubierta exterior de componentes está montada con vigas de madera de 1.5 cm de lado. Con todo esto, las dimensiones finales de la maqueta son de 70 cm de ancho, 45 cm de largo y 100 cm de alto.

Por último, la cubierta del invernadero que cubrirá el exterior de la instalación será de policarbonato. Se trata de un tipo de plástico con buena resistencia pero que los agentes naturales pueden degradar sus características correspondientes al paso de la radiación solar. Por este motivo, se trata de una cubierta con una esperanza de vida útil algo reducida si no se cuida correctamente.

7. Verificación

Una vez hecho el diseño se van a realizar algunas pruebas para comprobar el correcto funcionamiento del circuito electrónico y del funcionamiento del programa.

En primer lugar, se van a medir las corrientes que usa cada uno de los actuadores, para verificar así que el control creado mediante transistores es el adecuado. En caso de un funcionamiento por debajo de lo esperado se modificará el circuito para optimizar el sistema.

Para las mediciones de los actuadores se van a plantear tres casos, una conexión directa, una conexión mediante el BJT y por último, una control mediante Arduino:

- En la conexión directa, el actuador está conectado a una fuente de alimentación de tensión fija, una fuente de la marca PROMAX, con una tensión fija de 5 V y una corriente máxima de 2 A. En esta conexión, el componente recibe la corriente necesaria desde la fuente, sin haber ningún limitador ni carga adicional.
- En el segundo caso, el actuador está conectado entre la fuente de alimentación y el colector del BD137. En este caso, la base del transistor estará conectada a la misma fuente de alimentación mediante la resistencia de base. Como se mostró en los esquemas anteriores, debido a la naturaleza de estos actuadores se tomar precauciones mediante la conexión de paralelo de un 1N5408. En el caso de la placa térmica de cerámica, la carga en el colector será el relé, con el propio actuador en sus contactores.
- Por último, se repetirá el segundo caso, cambiando la alimentación del transistor de la fuente por una salida digital del Arduino, configurada a un estado alto, con una tensión de 5 V.

Se van a realizar las tres medidas en cada actuador, colocando un amperímetro en serie para la medición real de consumo.

Así pues, en el caso de la bomba se hace una medición adicional, algo equivalente a la carga que supondría el transporte de agua para el riego [Tabla 7].

***Tabla** 7 – Tabla informativa de los valores nominales de corriente obtenidos en los diferentes casos de conexión, además de una medida máxima de carga*

Mini bomba			
Corriente (mA)	Conexión directa	Conexión con BJT	Conexión con Arduino
Nominal	270	270	260
Con carga	600	450	445

Se ha aplicado también una carga, dificultando el paso mediante la presión con un dedo en el conducto de salida para simular una posible carga. De estas medidas se ha tomado un valor máximo observado, demostrando así que la resistencia calculada anteriormente podría no proporcionar la corriente necesaria en la base para saturar correctamente. Por esto motivo se modifica el valor de la resistencia para no disminuir el rendimiento de la bomba [Tabla 8].

Tabla 8 *– Corriente de base recalculada con el valor redimensionado de resistencia de base*

Actuador	V_{do} (V)	$R_{b\text{-}norm}$ (kΩ)	I_b (mA)
Bomba de agua	5	270	18.5 mA

Para el resto de actuadores se hacen las mismas medidas [Tabla 9], pero ninguna de estos tendrá una carga que suponga un valor que merezca ser medido.

Tabla 9 *– Tabla de resultados de las diferentes corrientes nominales de consumo medidas*

Actuador	Conexión directa (mA)	Conexión con BJT (mA)	Conexión con Arduino (mA)
Válvula	130	140	130
Ventilador	81	80	80
MCH	1200	1230	1220

A estos valores se les puede sumar la corriente máxima de consumo que tiene un Arduino NANO (I_{NANO}), 800 mA según especificaciones, para obtener la corriente máxima del sistema (I_{Tmax}):

$$I_{Tmax} = I_{NANO} + I_{Ventiladores} + I_{Valvulas} + I_{MCH} + I_{BombaMax}$$

$$I_{Tmax} = 800\ mA + (2 \cdot 80\ mA) + (2 \cdot 130\ mA) + 1220mA + 600\ mA = 3040\ mA$$

Para un máximo de 3.04 A, se va a utilizar un adaptador de corriente AC/DC modelo YU-0510, para tener una fuente de alimentación de 5 V 10 A para alimentar todo el sistema. Cabe mencionar que el cálculo es con todos los componentes activos, una situación no contemplada en el programa realizado debido a lógica impuesta, donde la placa calefactora no va a funcionar a la vez que los ventiladores.

Tras el montaje de la estructura y la instalación de los componentes (Figura 37), se procede a hacer pruebas del funcionamiento de Arduino. Para las pruebas de consumo se ha verificado el funcionamiento individual de los actuadores, por lo que falta comprobar la recepción de los sensores. Teniendo en cuenta el funcionamiento del sistema, se va a comprobar la conexión del Arduino al HMI, verificando así la lectura de los sensores y la comunicación serial a la vez.

Cabe mencionar que el montaje final difiere del diseño del plano en la cobertura exterior hecha para alojar componentes, que se ha movido a un lado para usar el apoyo vertical. Tampoco se ha puesto la cobertura esta parte para poder manejar correctamente el circuito en este periodo de pruebas.

***Figura 37** – Montaje realizado de la maqueta de invernadero. Se observan las dos plantas en el interior y todo el circuito electrónico en la parte exterior derecha, donde falta una cobertura para la protección contra la lluvia.*

De los tres tipos de sensores disponibles, se nota el mal funcionamiento de uno, el sensor de humedad, marcando una humedad máxima en todo momento. Tras la separación del resto del sistema se hacen las siguientes pruebas con los siguientes resultados:

- El sensor marca una humedad del 75.6 % en un suelo seco, lo equivalente a un valor de analógico de 250 sobre 1023, el rango de valores del sensor.
- En un recipiente lleno de agua marca una humedad del 68 %, con un valor analógico de 320 sobre 1023.

- Al aire libre, lo equivalente a lo más seco, marca un valor de humedad de 0%, lo equivalente a un valor analógico de 1023 sobre 1023.

Con estos resultados, se concluye que el sensor está dañado, marcando valores con un recorrido ilógico. En caso de marcar una humedad del 100 % al aire libre, se podría modificar el rango de valores útiles a un rango de [68 – 100] %, con una unidad invertida. Sin embargo, se perdería resolución en la medida.

Pese a la lectura errónea del sensor, la bomba se enciende y riega en las plantas adecuadas con el sensor al aire libre, marcando una humedad del 0 %. Junto a la bomba, el resto de los actuadores ejecutan la orden indicada dependiendo de las medidas.

Tras horas de pruebas se observan también los siguientes datos es referencia al control de temperatura:

- La temperatura interior del invernadero asciende a más de 30 ºC al estar bajo el sol, ya que no cuenta con una cubierta mecanizada. Estos valores son bastante perjudiciales para las plantas alojadas en el interior.
- La ventilación asistida mediante los ventiladores no tiene la capacidad de refrigerar la estancia pese a horas de funcionamiento.

Además del funcionamiento electrónico del sistema, se observa también la comunicación con el HMI. La conexión es correcta y el flujo de información no presenta ningún problema. Sin embargo, se muestran en la interfaz gráfica los problemas mencionados anteriormente (Figura 38).

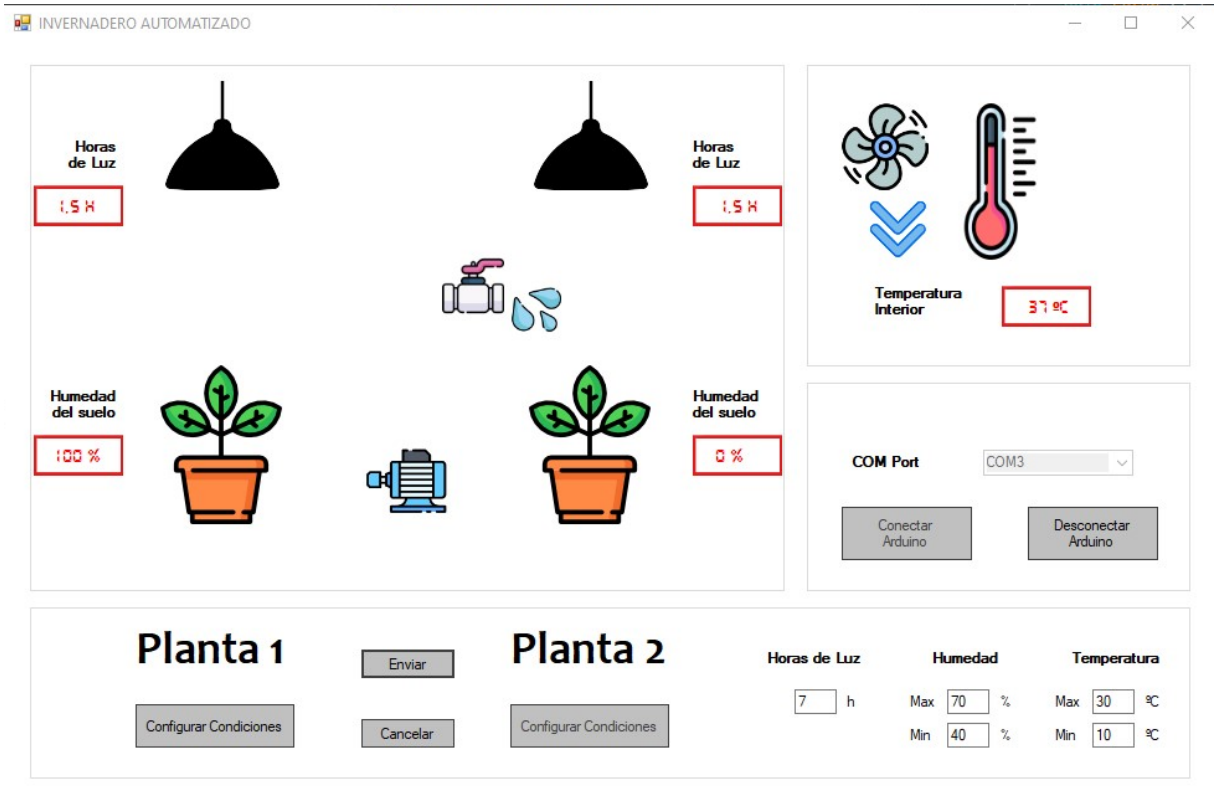


Figura 38 *– Imagen de la interfaz gráfica con los valores recibidos desde el Arduino mostrados en sus respectivas localizaciones, junto a su iconografía necesaria para representar el estado activo de los actuadores.*

Puesto que el sensor de humedad continúa marcando un valor de humedad de 100 o 0, se ha desconectado físicamente el conector de la bomba de agua, para no regar en exceso en la planta que tiene el sensor de humedad al aire.

La escritura de la configuración desde el HMI es viable, sin embargo, hay unos valores base en el código de configuración del Arduino, por lo que cada vez que se reinicia el programa hay que modificar esos valores en caso de ser lo deseado.

8. Conclusiones

Una vez realizadas las pruebas de funcionamiento después del montaje, se puede llegar a unas conclusiones, teniendo en cuenta la tecnología y los métodos mencionados a lo largo de este proyecto.

En primer lugar, El correcto funcionamiento necesita de un dimensionado adecuado para alcanzar un grado de control en cada variable del sistema. Con los componentes con los que se ha trabajado en este proyecto se han alcanzado limitaciones que hacen inviable la automatización completa de esta maqueta.

El grado de refrigeración no ha sido suficiente, por lo que se necesitaría de la mejora de capacidad de la tecnología usada o de una ayuda manual para combatir la temperatura alcanzada bajo el sol en la ubicación del montaje. De cara a una ampliación de este sistema, la implementación de un sistema de cobertura para parar el sol es fundamental. Otras opciones podrían ser el cambio de los métodos de control de temperatura, como podría ser la de ventilar aire refrigerado, no solo utilizar el aire ambiente exterior.

De estas mismas limitaciones se puede observar también la retención del calor producido por la absorción de rayos solares sobre las plantas y el suelo del invernadero. Esto demuestra la importancia de las características del material usado, en este caso de la cubierta, que puede aprovechar mejor la temperatura de las pocas horas de sol en un clima o estación más fríos.

En segundo lugar, el control diseñado puede funcionar con componentes de mayor potencia, ya que se usa la misma tecnología de control mediante transistores y relés independientemente de las corrientes y tensiones utilizadas por los componentes. Sin embargo, el número de componentes utilizados es casi el límite del procesador utilizado, por lo que una mayor precisión mediante un mayor número de sensores o actuadores significa también la necesidad de otro procesador de mayor capacidad de entradas y salidas.

En caso de una ampliación de espacio también se puede pensar en la conexión de varios procesadores a un ordenador central mediante una red de comunicaciones, controlando cada espacio mediante la misma interfaz gráfica.

Por último, tras la realización del diseño e instalación de un invernadero a pequeña escala se ha podido observar también el coste que supone un invernadero a una escala mayor, teniendo como objetivo una producción agrícola sostenible y eficiente. Con esto en mente, la contribución al ODS de hambre cero de este proyecto podría ser inviable, ya que la inversión necesaria para un aumento de producción en áreas del mundo donde falta más comida supone un gran costo. Aun así, la finalidad principal de proteger los cultivos ayuda a que la situación no empeore, paliando los efectos negativos del cambio climático en el sector agrícola mundial.

9. Referencias

[1] Serrano Cermeño, Zoilo: «Construcción de Invernaderos 2º Edición», 2002.

[2] https://www.cursosaula21.com/modbus-que-es-y-como-funciona/ (accedido el

09/7/2023)

[2] https://www.emb.cl/electroindustria/articulo.mvc?xid=1399&ni=wirelesshart-extendiendo-las-capacidades-del-protocolo-hart (accedido el

09/7/2023)

[4] https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf

[5] https://angelmicelti.github.io/VilladiegoSTEAM/invernadero/sensor-de-humedad-de-suelo-fc28.pdf

[6] https://servicio.mapa.gob.es/websiar/NecesidadesHidricas.aspx (accedido el

11/7/2023)

[7] http://www.seaflo.com/en-us/product/detail/604.html

[8] https://www.velleman.eu/products/view/?id=459188

[9] https://www.directindustry.es/prod/robert-bosch-gmbh/product-98211-2371093.html

[10] K.J. McCree: «The measurement of photosynthetically active radiation», 1971.

[11] https://pdf1.alldatasheet.es/datasheet-pdf/view/16170/PHILIPS/BD137.html

[12] https://www.mouser.com/datasheet/2/149/fairchild%20semiconductor_1n5408-543967.pdf

[13] https://www.farnell.com/datasheets/1699475.pdf

[15] https://www.handsontec.com/dataspecs/I2C_2004_LCD.pdf

ANEXOS

Código Arduino

```
//-------Librerias-------
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>
#include <DHT_U.h>
#include <EEPROM.h>
#include "Planta.h"


//-------Pines de entrada-------
int DHTPin = 2;
#define DHTTYPE DHT11
int Hum1Pin = A0;
int Hum2Pin = A1;
int LdrPin = 13;
```

```
//-------Pines de salida-------
int Valv2Pin = 12; //Al momento de crear los objetos de Planta se usara
este pin como ID
int Valv1Pin = 11; //Al momento de crear los objetos de Planta se usara
este pin como ID
int BombaPin = 10;
int HeatPin = 9;
int Fan2Pin = 8;
int Fan1Pin = 7;
int Led2Pin = 6;
int Led1Pin = 5;
//int CoverPin = 4; //Pin para cobertura en caso de haber


//-------Definicion de variables y objetos-------
int cont5s;
int cont10m;
int dhtAux;
int ldrAux;
float TempMedia;
String lcdRow1 = "";
String lcdRow2 = "";
String lcdRow3 = "";
String lcdRow4 = "";
bool action = false;
String output_HMI = "#H0A0B0C0D000"; //Formato inicial del envio al HMI
por comunicacion serial
String input_HMI = "";
bool inputDone = false;

LiquidCrystal_I2C lcd (0x27, 20, 4); //LCD de 20 columnas y 4 lineas
DHT dht (DHTPin, DHTTYPE);
Planta Planta1 (Valv1Pin);
Planta Planta2 (Valv2Pin);


//-------Funcion de configuracion-------
void setup () {
  Serial.begin(9600);

  cont5s = 0;
  cont10m = 0;
  valoresMedios ();

  pinConfig ();
  lcd.begin ();
  lcd.backlight ();
```

```
  dht.begin ();

  //Orégano (8h, 25 grados, 15 grados mínimos, 60 % humedad max, 45
%humedad min)
  Planta1.SetPlanta (48, 25, 15, 60, 45);
  //Cilantro (6h en unidad de 10min, 25 grados máximos, 10 grados, 60 %,
45 %)
  Planta2.SetPlanta (36, 25, 10, 60, 45);

}


//-------Bucle principal-------
void loop () {
  //En caso de haber una instruccion desde el HMI
  if (inputDone == true) {
    input_HMI.trim ();
    //serial.println(input_HMI); //Unicamente para comprobaciones con el
monitor serial de IDE
    serialConfig ();

    input_HMI = "";
    inputDone = false;
  }

 //ComprobaciÓn del estado de las plantas cada 5 segundos
  delay (4000); //El segundo que falta está más adelante
  //Almacenamiento y conversión de datos mediante los sensores
  dhtAux = int (dht.readTemperature ());
  ldrAux = contador (digitalRead (LdrPin));
  //Planta 1
  Planta1.ContadorLuz (ldrAux); //Comprueba si hace falta más luz
artificial
  Planta1.ComprobarAgua (analogRead (Hum1Pin)); //Comprueba si hace falta
riego
  Planta1.ComprobarTemp (dhtAux); //Comprueba si la temperatura es
correcta
  //Planta 2
  Planta2.ContadorLuz (ldrAux);
  Planta2.ComprobarAgua (analogRead (Hum2Pin));
  Planta2.ComprobarTemp (dhtAux);

  //Actualizacion del LCD
  if (action == false) lcdDataUpdate (); // Actualización de datos de las
plantas
  lcdUsage (); //Escritura de las strings actual en el lcd

  //Envio de trama al HMI
  serialOutputDHT ();
```

```
  Serial.println (output_HMI);
  Serial.flush ();
  delay (1000);
}


//-------Funciones auxiliares. Control de bomba-------
void RiegoOn (int n){
  if (n == Valv1Pin){
    if (digitalRead (BombaPin) == LOW){
      digitalWrite (Valv2Pin, HIGH);
      while (digitalRead (Valv2Pin) == LOW) {}
      digitalWrite (BombaPin, HIGH);

      lcdActionsUpdate ("ValvOn", 1);
      output_HMI.setCharAt (2, '1');
      action = true;
    }
    else {
      if (digitalRead (Valv1Pin == HIGH)) {
        digitalWrite (Valv1Pin, LOW);

        lcdActionsUpdate ("ValvOn", 1);
        output_HMI.setCharAt (2, '1');
        action = true;
      }
    }
  }
  else{
    if (digitalRead(BombaPin == LOW)) {
      digitalWrite (Valv1Pin, HIGH);
      while (digitalRead (Valv1Pin) == LOW) {}
      digitalWrite (BombaPin, HIGH);

      lcdActionsUpdate ("ValvOn", 1);
      output_HMI.setCharAt (2, '1');
      action = true;
    }
    else {
      if (digitalRead (Valv2Pin == HIGH)) {
        digitalWrite (Valv2Pin, LOW);

        lcdActionsUpdate ("ValvOn", 3);
        output_HMI.setCharAt (4, '1');
        action = true;
      }
    }
  }
}
```

```
void RiegoOff (int n){
  if (n == Valv1Pin){
    if (BombaPin == HIGH){
      if (digitalRead (Valv2Pin == LOW)) {
        digitalWrite (Valv1Pin, LOW);

       lcdActionsUpdate ("ValvOff", 1);
       output_HMI.setCharAt (2, '0');
       action = true;
      }
      else {
        digitalWrite (BombaPin, LOW);
        while (digitalRead (BombaPin) == HIGH) {}
        digitalWrite (Valv2Pin, LOW);
        digitalWrite (Valv1Pin, LOW);

        lcdActionsUpdate ("ValvOff", 1);
        output_HMI.setCharAt (2, '0');
        action = true;
      }
    }
  }
  else {
    if (BombaPin == HIGH){
      if (digitalRead (Valv2Pin == LOW)) {
        digitalWrite (Valv2Pin, LOW);

       lcdActionsUpdate ("ValvOff", 3);
       output_HMI.setCharAt (4, '0');
       action = true;
      }
      else {
        digitalWrite (BombaPin, LOW);
        while (digitalRead (BombaPin) == HIGH) {}
        digitalWrite (Valv2Pin, LOW);
        digitalWrite (Valv1Pin, LOW);

        lcdActionsUpdate ("ValvOff", 3);
        output_HMI.setCharAt (4, '0');
        action = true;
      }
    }
 }
}

//-------Funciones auxiliares. Control de ventilador-------
void FanOn (int n) {
  digitalWrite (Fan1Pin, HIGH); //Cambia el estado de un ventilador
```

```
  digitalWrite (Fan2Pin, HIGH);

  if (n == Valv1Pin) lcdActionsUpdate ("Tempv", 1);
  else lcdActionsUpdate ("Tempv", 2);
  output_HMI.setCharAt (6, '1');
  action = true;
}

void FanOff (int n) {
  digitalWrite (Fan1Pin, LOW);
  digitalWrite (Fan2Pin, LOW);

  if (n == Valv1Pin) lcdActionsUpdate ("Temp-", 1);
  else lcdActionsUpdate ("Temp-", 2);
  output_HMI.setCharAt (6, '0');
  action = true;
}


//-------Funciones auxiliares. Control de calefactor-------
void HeatOn (int n) {
  digitalWrite (HeatPin, HIGH); //Cambia el estado del calefactor

  if (n == Valv1Pin) lcdActionsUpdate ("Temp^", 1);
  else lcdActionsUpdate ("Temp^", 2);
  output_HMI.setCharAt (6, '2');
  action = true;
}

void HeatOff (int n) {
  digitalWrite (HeatPin, LOW);

  if (n == Valv1Pin) lcdActionsUpdate ("Temp-", 1);
  else lcdActionsUpdate ("Temp-", 2);
  output_HMI.setCharAt (6, '0');
  action = true;
}


//-------Funcion auxiliar. Control de luces-------
void LedOn (int n) {
  if (n == Valv1Pin) {
    if (digitalRead (Led1Pin == LOW)) {
      digitalWrite (Led1Pin, HIGH); //Cambia el estado de los leds de una
planta

      lcdActionsUpdate ("LEDOn", 1);
      output_HMI.setCharAt (8, '1');
      action = true;
```

```
    }
  }
  if (n == Valv2Pin) {
    if (digitalRead (Led1Pin == LOW)) {
          digitalWrite (Led2Pin, HIGH);

      lcdActionsUpdate ("LEDOn", 2);
      output_HMI.setCharAt (10, '1');
      action = true;
    }
  }
}

void LedOff (int n) {
  if (n == Valv1Pin) {
    if (Led1Pin == HIGH) {
      digitalWrite (Led1Pin, LOW);

      lcdActionsUpdate ("LEDOff", 1);
      output_HMI.setCharAt (8, '0');
      action = true;
    }
  }
  if (n == Valv2Pin) {
    if (Led2Pin == HIGH) {
      digitalWrite (Led2Pin, LOW);

      lcdActionsUpdate ("LEDOff", 2);
      output_HMI.setCharAt (10, '0');
      action = true;
    }
  }
}


//-------Funcion auxiliar. Visualizacion con LCD-------
void lcdUsage () {
  lcd.clear ();

  //Imprime las cadenas de carcteres en el LCD una por una
  lcd.setCursor (0, 0);
  lcd.print (lcdRow1);
  lcd.setCursor (0, 1);
  lcd.print (lcdRow2);
  lcd.setCursor (0, 2);
  lcd.print (lcdRow3);
  lcd.setCursor (0, 3);
  lcd.print (lcdRow4);
```

```
  //Reinicia las cadenas de caracteres
  lcdRow1 = "";
  lcdRow2 = "";
  lcdRow3 = "";
  lcdRow4 = "";
  action = false; //Reinicia variable auxiliar de cambio de estado en los
actuadores
}

void lcdDataUpdate () {
  int n;
  if (cont5s & 0x1) { //Cada 5s cambia entre Planta 1 y Planta 2
    lcdRow1 = "PLANTA 1 - OREGANO";
    lcdRow2 = "";
    lcdRow3 = " Temp | H Luz | Hum ";
    //Prepara la cuarta cadena, transformando los valores actuales a una
unidad representable
    lcdRow4 = " ";
    lcdRow4.concat (dhtAux);
    lcdRow4.concat ("ºC | ");
    n = int ((Planta2.AuxLuz / 6) * 100);
    lcdRow4.concat (n / 100);
    lcdRow4.concat ("h | ");
    n = (Planta2.AuxAgua);
    lcdRow4.concat (n);
  }

  else {
    lcdRow1 = "PLANTA 2 - CILANTRO";
    lcdRow2 = "";
    lcdRow3 = " Temp | H Luz | Hum ";
    lcdRow4 = " ";
    lcdRow4.concat (dhtAux);
    lcdRow4.concat ("ºC | ");
    n = int ((Planta1.AuxLuz / 6) * 10);
    lcdRow4.concat (n / 10);
    lcdRow4.concat ("h | ");
    n = (Planta1.AuxAgua);
    lcdRow4.concat (n);
  }
}

void lcdActionsUpdate (String c, int n) {
  //Solo entra si se ha encendido o apagado algun actuador
  lcdRow1 = "PLANTA 1 - OREGANO";
  lcdRow3 = "PLANTA 2 - CILANTRO";
  if (n == 1){
    lcdRow2.concat (c);
    lcdRow2.concat (" ");
```

```
  }
  else {
    lcdRow4.concat (c);
    lcdRow4.concat (" ");
  }
}


//-------Funcion auxiliar. Configuracion de pines-------
void pinConfig () {
  pinMode (Valv1Pin, OUTPUT);
  pinMode (Valv2Pin, OUTPUT);
  pinMode (Fan1Pin, OUTPUT);
  pinMode (Fan2Pin, OUTPUT);
  pinMode (BombaPin, OUTPUT);
  pinMode (HeatPin, OUTPUT);
  pinMode (Led1Pin, OUTPUT);
  pinMode (Led2Pin, OUTPUT);
  pinMode (LdrPin, INPUT);
}


//-------Funcion auxiliar. Calculo de valores medios de funcionamiento---
----
void valoresMedios () {
  int n1, n2;
  if (Planta1.LeerTempMin () > Planta2.LeerTempMin ()) n1 =
Planta1.LeerTempMin ();
  else n1 = Planta2.LeerTempMin ();

  if (Planta1.LeerTempMax () < Planta2.LeerTempMax ()) n1 =
Planta1.LeerTempMax ();
  else n1 = Planta2.LeerTempMax ();

  TempMedia = (n1 + n2) / 2;
}


//-------Funcion auxiliar. Contador de horas de luz-------
int contador (int n) {
  if (Led1Pin == HIGH | Led2Pin == HIGH) if (n == 0) n = 1;
  //Se considera q hay luz si los leds estan encendidos pero el ldr no lo
capta por capacidad
  //Se mide el tiempo en unidad de 5 segundos y de 10 minutos para las
horas de luz
  if (n == 1) {
    cont5s++;
    if (cont5s == 120) {
      cont10m++;
```

```
      cont5s = 0;
      if (cont10m == 144) cont10m = 0;
      return 1;
    }
    else return 0;
  }
  else return 0;
}


//-------Definicion de funciones de la clase Planta-------
Planta::Planta (int n){
  ID = n;
}

void Planta::SetPlanta (int h, int tmax, int tmin, int humax, int humin)
{
  LuzHoras = h;
  TempMax = tmax;
  TempMin = tmin;
  HumMax = humax;
  HumMin = humin;
  AuxLuz = 0; AuxAgua = 0; AuxTemp = 0;
}

void Planta::ContadorLuz (int n) {
  if (n == 1) AuxLuz++; //Suma una vez cada 10 minutos de incidencia de
luz
  if (AuxLuz == 144) AuxLuz = 0; //Reinicia el contador cada 24h

  if (n == 1) { //En caso de haber luz
    LedOff (ID); //Se llama a la función que apaga los leds asociados a
esta planta
    //En caso de haber una cobertura para sombra se comprobaría si hay
exceso de luz
  }
  else if (AuxLuz < LuzHoras) { //En caso de necesitar más luz
    LedOn (ID); //Se llama a la función que enciende los leds asociados a
esta planta
  }
}

void Planta::ComprobarAgua (int hum) {
  AuxAgua = int (100 * (1 - (hum / 1023))); //Se añade la humedad actual
a la variable auxiliar en un formato de porcentaje

  if (AuxAgua < HumMin) RiegoOn (ID); //Enciende el riego de esta planta
  else if (AuxAgua > ((HumMax + HumMin)/2) && digitalRead (ID) != LOW)
RiegoOff (ID);
```

```
  //Se desea llegar a un valor óptimo, no entrar unicamente en el rango
}

void Planta::ComprobarTemp (int temp) {
  AuxTemp = temp; //Almacena el valor de temperatura actual

  //Comprueba si necesita ventilación mediante una media de temperatura
de ambas plantas
  if (temp > TempMax) { if (digitalRead (Fan2Pin == LOW)){ FanOn (ID);} }
  else if (temp < TempMedia && digitalRead (Fan1Pin) != LOW) FanOff (ID);
  //Comprueba si necesita calefacción mediante una media de temperatura
de ambas plantas
  if (temp < TempMin) { if (digitalRead (HeatPin == LOW)) HeatOn (ID); }
  else if (temp > TempMedia && digitalRead (HeatPin) != LOW) HeatOff
(ID);
}

int Planta::LeerLuz () {
  return LuzHoras;
}

int Planta::LeerHumMax () {
  return HumMax;
}

int Planta::LeerHumMin () {
  return HumMin;
}

int Planta::LeerTempMin () {
  return TempMin;
}

int Planta::LeerTempMax () {
  return TempMax;
}


//-------Funciones auxiliares. Comunicacion serial-------
void serialEvent () {
  //Funcion llamada al final de cada iteracion del loop
  while (Serial.available ()){ //Funcionaen caso de haber un menasje
entrante
    char inputChar = Serial.read (); //Variable auxiliar que lee cada
byte
    input_HMI += inputChar; //Se añade cada byte leido en forma de
caracteres en una cadena global
```

```
    if (inputChar == '\n') inputDone = true; //Se indica que se ha
finalizado la lectura
  }
}

void serialConfig () {
  //Configuracion de los valores de crecimiento de cada planta desde el
HMI
  String n[6]; //Se crea una matriz de strings auxiliares
  int i = 0; //Variable auxiliar

  if (input_HMI.indexOf ("#A") != -1) { //Correspondiente a la planta 1
    input_HMI = input_HMI.substring (2); //Se elimina el identificativo
    //Se separan de la cadena cada uno de los valores separados por comas
    while (input_HMI.indexOf (",") != -1) {
      i++;
      n[0] = input_HMI.substring (0, input_HMI.indexOf (","));
      input_HMI = input_HMI.substring (input_HMI.indexOf (",") + 1);
      n[i] = n[0].toInt ();
    }
    n[5] = input_HMI.toInt ();
    //Se configuran los nuevos valores de la planta 1
    Planta1.SetPlanta (n[1].toInt (), n[2].toInt (), n[3].toInt (),
n[4].toInt (), n[5].toInt ());
  }
  else if (input_HMI.equals("#B") !=1) { //Mismo proceos anterior pero en
la planta 2
    input_HMI = input_HMI.substring (2);
    while (input_HMI.indexOf (",") != -1) {
      i++;
      n[0] = input_HMI.substring (0, input_HMI.indexOf (","));
      input_HMI = input_HMI.substring (input_HMI.indexOf (",") + 1);
      n[i] = n[0].toInt ();
    }
    n[5] = input_HMI.toInt ();
    Planta2.SetPlanta (n[1].toInt (), n[2].toInt (), n[3].toInt (),
n[4].toInt (), n[5].toInt ());
  }
}

void serialOutputDHT () {
  output_HMI.remove (11);
  output_HMI.concat (dhtAux);
  output_HMI.concat (",");
  output_HMI.concat (Planta1.AuxAgua);
  output_HMI.concat (",");
  output_HMI.concat (Planta2.AuxAgua);
  output_HMI.concat (",");
  output_HMI.concat (int ((Planta1.AuxLuz / 6) * 10));
```

```
  output_HMI.concat (",");
  output_HMI.concat (int ((Planta2.AuxLuz / 6) * 10));
}
```

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

Grado en Ingeniería Electrónica Industrial y Automática

Curso 2022/2023

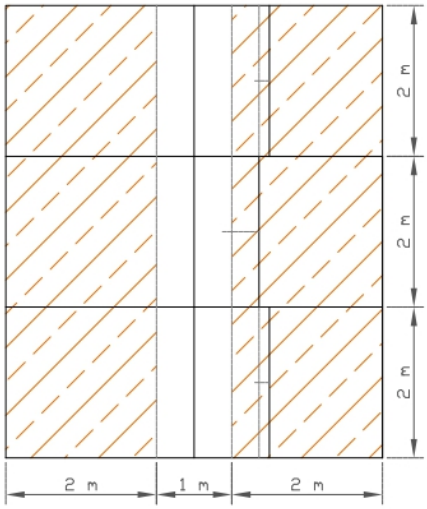
TRABAJO DE FIN DE GRADO:

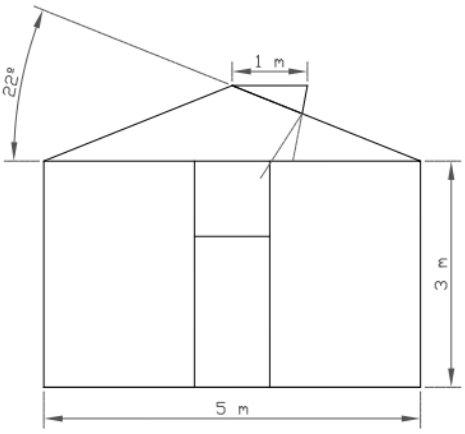
CONTROL ELECTRÓNICO DE UN INVERNADERO

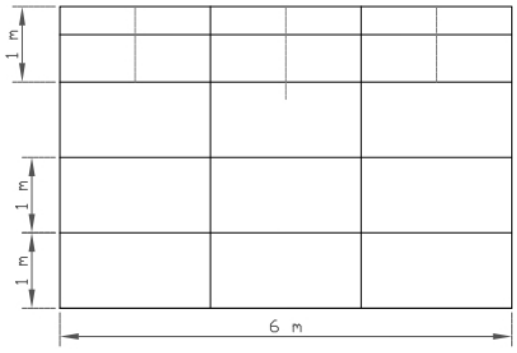
DOCUMENTO 2. PLANOS

AUTOR: Adair Enrique Cedeño Vinces

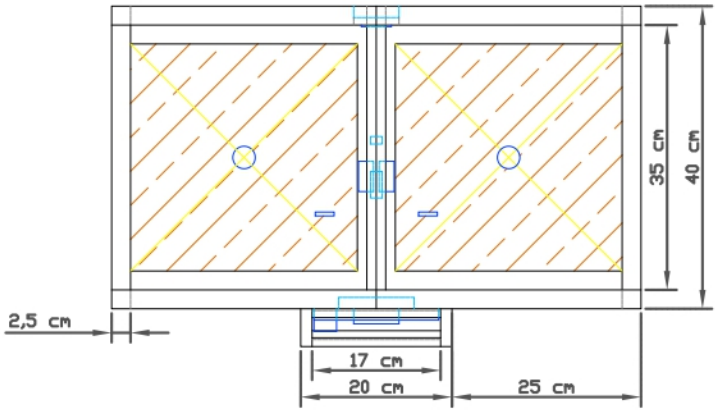
TUTOR: Dr. Enrique Berjano Zanón

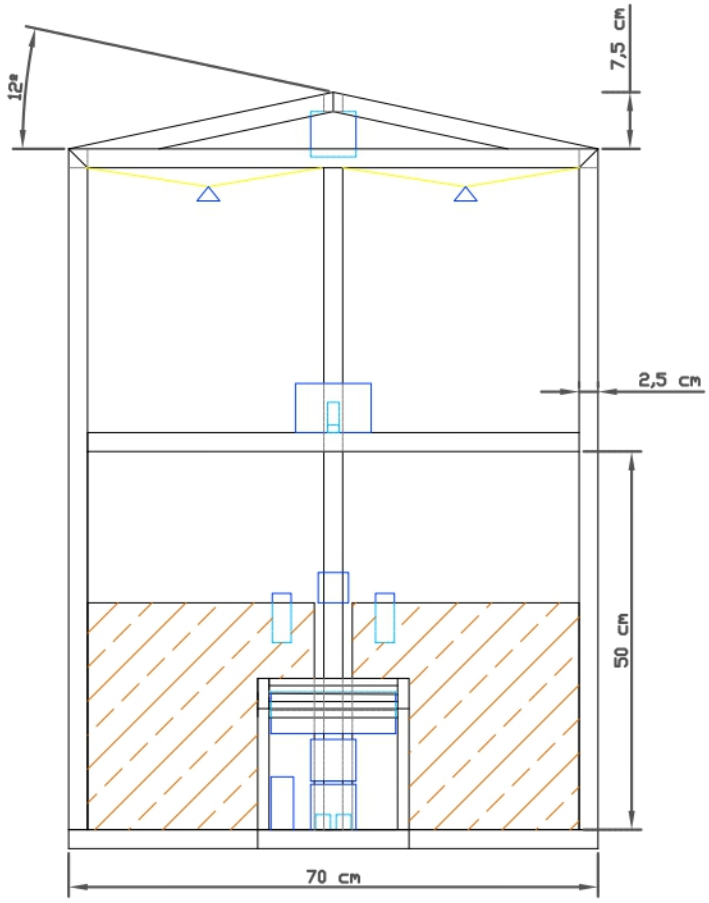


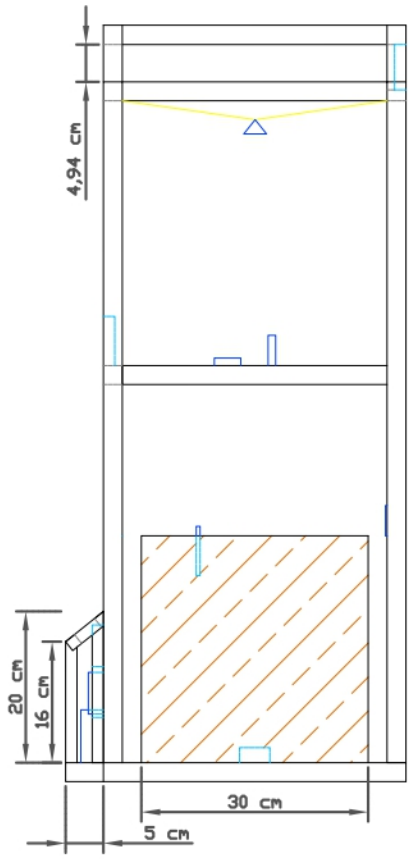




	Fecha	Nombre	UNIVERISDAD POLITÉCNICA DE VALENCIA
Dibujado	12/07/2023	Adair Enrique Cedeño Vinces	
Comprobado	-	-	ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO
Escala: 1:100	Estructura exterior de un invernadero tipo capilla a dos aguas		Curso: 2022/2023





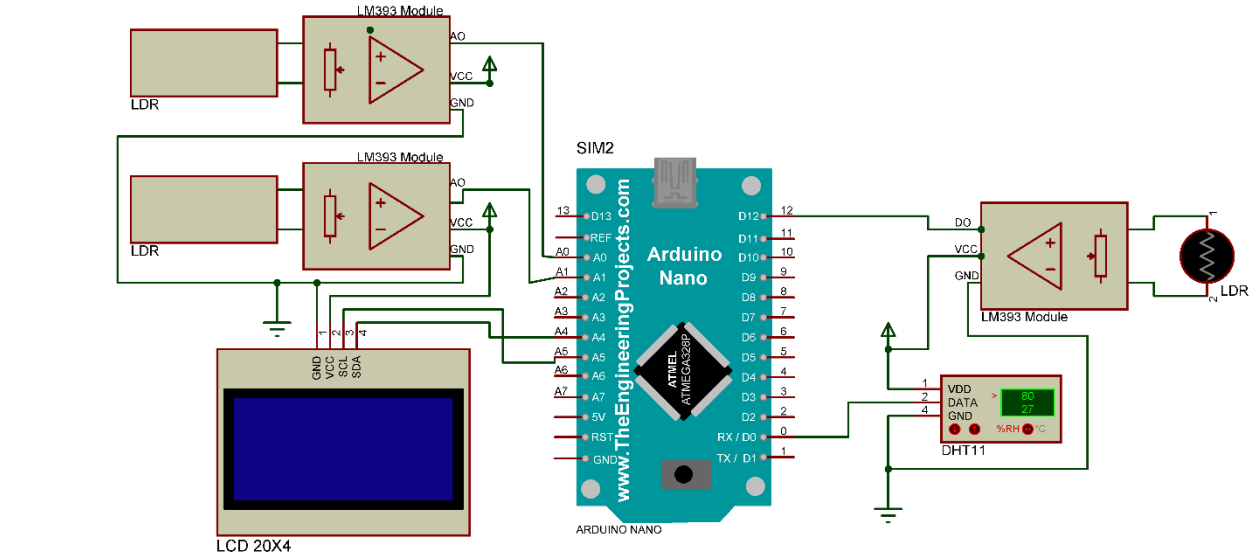


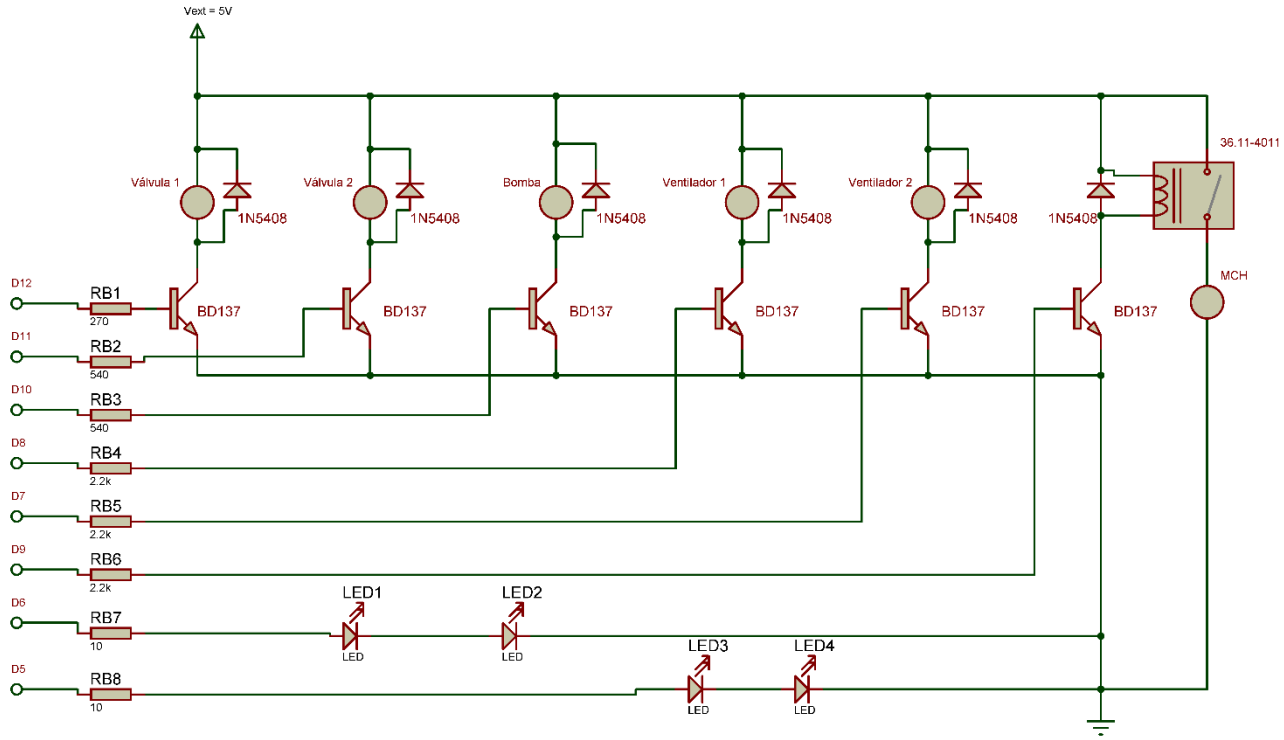
	Fecha	Nombre
Dibujado	12/07/2023	Adair Enrique Cedeño Vinces
Comprobado	-	-
Escala: 1:10	Estructura exterior y posición de componentes de la maqueta de invernadero	

UNIVERISDAD POLITÉCNICA DE VALENCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

Curso: 2022/2023





	Fecha	Nombre	UNIVERSIDAD POLITÉCNICA DE VALENCIA
Dibujado	15/07/2023	Adair Enrique Cedeño Vinces	
Comprobado	-	-	ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO
Escala:	Circuito completo y rectificado del sistema basado en Arduino		Curso: 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:

CONTROL ELECTRÓNICO DE UN INVERNADERO

DOCUMENTO 3. PLIEGO DE CONDICIONES

AUTOR: Adair Enrique Cedeño Vinces

TUTOR: Dr. Enrique Berjano Zanón

1. Objeto

El presente documento tiene la finalidad de especificar los criterios para el desarrollo y montaje de una maqueta de invernadero basada en un modelo a escala real.

2. Requisitos

Con el fin de obtener un producto final se necesitan de dos apartados fundamentales a implementar, los componentes que en conjunto suman el montaje y los requisitos técnicos de funcionamiento del sistema. Además, se han de cumplir con las normativas presentes correspondientes a la instalación de invernaderos

2.1. Especificaciones de diseño de software

La maqueta de invernadero ha de cumplir con su función de proteger una serie de cultivos, así como monitorizar el sistema de manera gráfica y manipular las variables de control de funcionamiento de la instalación. Así pues, los requisitos por parte del programa son los siguientes:

- Monitorización continua del estado de la instalación.
- Manipulación del sistema desde una interfaz gráfica dirigida al usuario.
- Control autónomo del sistema sin la necesidad de intervención humana.

2.2. Lista de materiales

Para un correcto funcionamiento del sistema hace falta de componentes para la medición y el control de las diferentes variables. A continuación, se listan los elementos necesarios para la instalación sin especificar función ni cantidad:

- Estructura exterior estable de madera de pino.
- Cubiertas de plástico con las características de transparencia y aislamiento térmico.
- Arduino NANO, procesador principal del sistema basado en ATmega328. Alimentación de 5 V con un consumo máximo de 800 mA.
- DHT11 del fabricante Aosong, sensor de temperatura y humedad relativa del aire, alimentado a 5 V con una resolución de 0.1 ºC y 1% de humedad relativa.
- FC-28, sensor de humedad del suelo junto a módulo comparativo basado en LM393, de la marca EstarDyn. El componente es alimentado a 5 V con una sensibilidad de 0.1 % de humedad.
- GL5539, fotorresistencia regulable mediante módulo comparativo basado en LM393, un amplificador operacional. El conjunto es de la marca Aoqdqdqd.
- Bomba de agua, modelo MYP3701 de la marca Yunt. Bomba de agua de presión negativa alimentada a 5 V y con un consumo nominal de 300 mA. Proporciona una presión máxima de 650 mmHg.
- Electroválvula de la marca Zonhen, modelo ZHV-0519L/S. Tiene una alimentación de 5V y una potencia nominal de 1.5 W.

- Ventilador de refrigeración 6010 CC de Caizu-fan, alimentado a 5 V con un consumo nominal de 60 mA. Gira a una velocidad de 1600 RMP, con un error de ±5 %.
- MCH, placa calefactora de cerámica de la marca Haljia, alimentada a 5 V con una potencia de consumo normalizada de 8 W.
- Led rojo 5 mm de la marca Fullwat, con una tensión de funcionamiento máxima de 2.4 V y una corriente de alimentación de 20 mA.
- Resistencias de la marca Velleman serie E-12, con una tolerancia de ±10 %.
- 1N5408 del fabricante Fairchild, diodo capaz de funcionar en conducción directa hasta un máximo de 3 A y 1000 V.
- BD137, semiconductor fabricado por la marca Philips. Transistor NPN con corriente de colector máxima de 1.5 A y tensión colector-emisor de 100 V.
- Relé de la marca Finder. Modelo 36.11.9.005.4011, perteneciente a la gama 36 SERIES. Soporta una tensión en corriente continua de hasta 48 V, además de poder ser utilizado en corriente alterna. En el contactor soporta una corriente máxima de 10 A.
- LCD de la marca GeeekPi, modelo IIC/I2C 2004. Display de 20x4, 20 filas y 4 columnas.
- Adaptador de corriente AC/DC de la marca Velain, modelo YU-0510. Transforma a una corriente de 5 V y 10 A de corriente continua.

2.3. Normativa vigente

Para la construcción de un invernadero se pueden aplicar las siguientes normativas referentes a la normalización española:

- UNE 13031-1:2020 / Invernaderos. Cálculo y construcción. Parte 1: Invernaderos para producción comercial.
- UNE 53328:1985 EX / plásticos. Películas de polietileno de baja densidad para invernaderos. Características y métodos de ensayo.
- UNE 76208:1992 EX / Estructuras metálicas. Invernaderos multicapilla con cubierta de materiales plásticos. Proyecto y construcción.

3. Prueba de servicio

Para corroborar el cumplimiento de los requisitos del proyecto se han de cumplir las siguientes pruebas en referencia a cada apartado:

3.1. Funcionamiento del programa

El sistema debe poder funcionar durante un periodo mínimo de un día para comprobar el control correcto de los sistemas del programa. Además, se debe poder manipular las variables del programa sin la desconexión y reinicio del sistema ni la manipulación directa del procesador.

3.2. Funcionamiento del circuito electrónico

Tras la instalación se debe comprobar el funcionamiento de todos los componentes en un circuito completo, asegurando el funcionamiento de todos dentro de las características eléctricas correspondientes a cada elemento de manera individual.

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:

CONTROL ELECTRÓNICO DE UN INVERNADERO

DOCUMENTO 4. PRESUPUESTO

AUTOR: Adair Enrique Cedeño Vinces

TUTOR: Dr. Enrique Berjano Zanón

El presente documento refleja los costes asociados a la completa instalación de la maqueta de un invernadero basado en un invernadero a escala real.

Se van a desglosar los costes referentes al material necesario para la instalación, el equipo electrónico usado, las licencias de software usadas para el diseño y los honorarios por mano de obra:

- Material de la instalación. Se incluyen todos los materiales referentes a la construcción y al diseño electrónico del proyecto.
- Equipo electrónico. Se incluyen los aparatos electrónicos usados para las pruebas del correcto funcionamiento del diseño electrónico.
- Licencias de software. Se incluyen las licencias de pago de los programas de software utilizados para el diseño del sistema. Se excluyen todos los softwares de carácter gratuito y de libro acceso.
- Mano de obra. Se incluye una estimación de las horas necesarias para la totalidad de este proyecto, así como el nivel de cualificaciones necesario dependiendo del apartado.

Tabla 10 *– Precios elementales del diseño e instalación del proyecto*

Cuadro de precios elementales			
Tipo	Fabricante	Descripción	Precio (€)
Materiales			
M1	-	Estructura de madera de pino	33.83 €
M2	-	Cubierta de plástico de policarbonato	35.78 €
M3	Arduino	Arduino NANO basado en microcontrolador ATmega328	21.6 €
M4	Aosong	Sensor de temperature y huemdad relativa del aire DHT11	2.14 €
M5	EstarDyn	Sensore de humedad en el suelo FC-28 junto a módulo comparativo basado en amplificadores operacionales LM393	0.6 €
M6	Aoqdqdqd	Fotorresistor GL5539 junto a módulo comparativo basado en amplificadores operacionales LM393	0.42 €
M7	Yunt	Bomba de vacío de presión MYP3701, de presión máxima de 650 mmHg	5.4 €
M8	Zonhen	Electroválvula ZHV-0519L/S normalmente abierta	2.83 €
M9	Caizu-fan	Ventilador de refrigeración 6010 CC, de una velocidad nominal de 1600 RPM	2.82 €
M10	Haljia	Placa calefactora de cerámica MCH, de 40 mm x 40 mm	12.99 €
M11	Fullwat	Led rojo de 5 mm, con referencia RF-WW05A3SRA1-B3	0.31 €

M12	Velleman	Kit de resistencias serie E-12 con una tolerancia de ±10 %	7.25 €
M13	Fairchild	Diodo rectificador de propósito general 1N5408	2.85 €
M14	Philips	Transistor NPN, modelo BD137	0.32 €
M15	Finder	Relé de la gama 36 SERIES, modelo 36.11.9.005.4011	2.26 €
M16	Geekpi	LCD, modelo IIC/I2C 2004, de 20 columnas por 4 filas	12.99 €
M17	Velain	Adaptador de corriente AC/DC de 5V y 10 A. Modelo YU-0510	20.99 €
Equipo electrónico			
E1	C-logic	Multímetro digital C-logic 520	19.99 €
E2	Promax	Fuente de alimentación regulable FAC-363	550 €
Software			
S1	Labcenter Electronics Ltd	Licencia de Proteus VSM Arduino AVR	260 €
Mano de obra			
O1	-	Ingeniero Técnico en Electrónica Industrial y Automática	20 €
O2	-	Técnico de instalación	12 €
Medios auxiliares sobre costos directos			
5%			

Tabla 11 *– Precios descompuestos del diseño e instalación del proyecto*

Cuadro de precios descompuestos					
Tipo	Fabricante	Descripción	Cantidad (ud)	Precio (€)	Importe (€)
Materiales					
M1	-	Estructura de madera de pino	1	33.83 €	33.83 €
M2	-	Cubierta de plástico	1	35.78 €	35.78 €
M3	Arduino	Arduino NANO basado en microcontrolador ATmega328	1	21.6 €	21.6 €
M4	Aosong	Sensor de temperature y huemdad relativa del aire DHT11	1	2.14 €	2.14 €

M5	EstarDyn	Sensor de humedad en el suelo FC-28 junto a módulo comparativo basado en amplificadores operacionales LM393	2	0.6 €	1.2 €
M6	Aoqdqdqd	Fotorresistor GL5539 junto a módulo comparativo basado en amplificadores operacionales LM393	1	0.42 €	0.42 €
M7	Yunt	Bomba de vacío de presión MYP3701, de presión máxima de 650 mmHg	1	5.4 €	5.4 €
M8	Zonhen	Electroválvula ZHV-0519L/S normalmente abierta	2	2.83 €	5.66 €
M9	Caizu-fan	Ventilador de refrigeración 6010 CC, de una velocidad nominal de 1600 RPM	2	2.82 €	5.64 €
M10	Haljia	Placa calefactora de cerámica MCH, de 40 mm x 40 mm	1	12.99 €	12.99 €
M11	Fullwat	Led rojo de 5 mm, con referencia RF-WW05A3SRA1-B3	4	0.31 €	1.24 €
M12	Velleman	Kit de resistencias serie E-12 con una tolerancia de ±10 %	1	7.25 €	7.25 €
M13	Fairchild	Diodo rectificador de propósito general 1N5408	6	2.85 €	17.1 €
M14	Philips	Transistor NPN, modelo BD137	6	0.32 €	1.92 €
M15	Finder	Relé de la gama 36 SERIES, modelo 36.11.9.005.4011	1	2.26 €	2.26 €
M16	Geekpi	LCD, modelo IIC/I2C 2004, de 20 columnas por 4 filas	1	12.99 €	12.99 €
M17	Velain	Adaptador de corriente AC/DC de 5V y 10 A, Modelo YU-0510	1	20.99 €	20.99 €
Equipo electrónico					
E1	C-logic	Multímetro digital C-logic 520	1	19.99 €	19.99 €
E2	Promax	Fuente de alimentación regulable FAC-363	1	550 €	550 €
Software					

S1	Labcenter Electronics Ltd	Licencia de Proteus VSM Arduino AVR	1	260 €	260 €
Mano de obra					
O1	-	Ingeniero Técnico en Electrónica Industrial y Automática	285 h	20 €/h	5700 €
O2	-	Técnico de instalación	15 h	12 €/h	180 €
Medios auxiliares sobre costos directos					
5%	6898.4 €	7243.32 €			

__Tabla 12__ – Resumen del presupuesto desglosado en secciones, con el porcentaje de medios auxiliares aplicado

Resumen del presupuesto		
Referencia	Descripción	Importe (€)
M	Materiales de la instalación	197.83 €
E	Equipo electrónico de prueba	598.49 €
S	Licencias de software	273 €
O	Mano de obra	6174 €
TOTAL PRESUPUESTO EJECUCIÓN MATERIAL		**7243.32 €**
13 %	Gastos generales	941.63 €
6 %	Beneficio industrial	434.59 €
PRESUPUESTO DE EJECUCIÓN POR CONTRATA		**8619.54 €**