_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE UPV INGENIEROS DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DISEÑO DE UNA ESTACIÓN DE SENSORES PARA LA MONITORIZACIÓN DE LAS MAGNITUDES FÍSICAS RELACIONADAS CON EL CRECIMIENTO DE UN CULTIVO

Sergio Cano León

Tutor: Ángel Héctor García Miquel

Cotutor: Vicente Torres Carot

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2015-16

Valencia, 1 de septiembre de 2016

“*En agradecimiento a todas aquellas personas que han confiado en mí, que han sabido cómo llevarme por el buen camino.*

A mis amigos, que con los momentos compartidos han llenado mi vida de felicidad y de dicha, consiguiendo mi sonrisa día a día.

A mis profesores, incluyendo desde primaria hasta mis últimos pasos en la universidad, que han formado al joven ingeniero dotándolo con la sabiduría y educación de la que hoy mi familia y amigos se sienten orgullosos.

A mis hermanos, mis defensores en vida; aquellos que han sido los pilares de mi existencia. Nunca podré olvidar todo lo que habéis hecho para ayudarme a llegar donde hoy estoy, el poder de los cinco nos acompaña, nunca lo olvidéis.

A mi padre, que a pesar de no poder comprender actualmente lo que escribo, pienso, digo o hago, sé que me hubiera apoyado en todo momento y, ahora mismo, estaría boyante de todo lo que acontece.

Por último, la más importante en mi vida, mi madre. La luchadora, la que ha dado todo por mis hermanos y por mí, la que cuida de mi padre día a día como si de su vida se tratase. La que llora cuando lloramos y ríe en días de gloria junto a nosotros. La que me ha permitido cumplir el sueño del que hoy es participe. Mi corazón, mi fuente de energía.”

Resumen

En la presente memoria del Trabajo Final de Grado realizado, se expone todos los conocimientos, tanto teóricos como prácticos, utilizados para el estudio, el diseño y la implementación de una estación de sensores encargada de la recolección de datos referentes a las magnitudes físicas que afectan al crecimiento/cuidado de un cultivo, tales como temperatura, humedad ambiental, humedad del terreno, presión atmosférica, niveles de iluminación (visible e infrarrojos) y color de la hoja/el fruto. Dicha recolección de datos, además del posterior envío de los mismos a una base de datos para su futura exportación a una aplicación Android, se realizará mediante la programación de la placa de circuito impreso Arduino.

Partiremos de unos estudios relativos al crecimiento del cultivo, lo que nos permitirá seleccionar los sensores más óptimos en relación a los objetivos a conseguir; se continuará con la descripción de la metodología empleada para la implementación de todo el sistema, exponiendo los resultados pertinentes, y, por último, se hablará de las conclusiones, así como de planes de mejora y futuras líneas de trabajo.

Palabras clave: sensores, red de sensores, comunicación inalámbrica, cultivo, monitorización cultivo.

Resum

En la present memòria del treball final de grau realizat, s'exposen tots el coneixements, tant teòrics com pràctics, utilitzats per a l'estudi, el diseny i la implementació d'una estació de sensors encarregada de la recolecció de dades referents a les magnituts físiques que afecten al creiximent d'un cultiu, com temperatura, humitat ambiental, humitat del terreny, pressió atmosférica, nivells d'il·luminació i color del full del fruit. Aquesta recolecció de dades a més del posterior enviament d'estos a una base de dades para la seua futura exportació a una aplicació Android, es realitzarà a través de la programació d'una placa de circuit imprés Arduino.

Partirem d'uns estudis relatius al creiximent del cultiu, que ens permetrà triar els sensors més òptims en relació als objectius a aconseguir, es continuarà amb la descripció de la metodologia utilitzada per a la implementació de tot el sistemes, exposant els resultats pertinents i, per últim, es parlarà de les conclusions, del plan de millora i futures llinees de treball.

Paraules claus: sensors, xarxa de sensor, comunicació inalàmbrica, cultiu, monotorització de cultiu.

Abstract

Herein, about the report of the work order degree completed, it exposed all knowledge, both theoretical and practical, used for the study, design and implementation of a sensor station in charge of data collection concerning physical quantities that affect to cultivation's growth/care, such as temperature, humidity (air), humidity (ground), atmospheric pressure, lighting levels (visible and infrared light) and leaf's color /fruit's color. This data collection, in addition to

sending them to a database for future export to an Android app, will be performed by programming printed circuit board Arduino.

We will start with studies on the cultivation's growth/care; it allows us to set the most optimal sensors in relation to the objectives to be achieved. It will continue with the description of the methodology used for the implementation of the whole system, exposing relevant results, and finally it will discuss the findings and improvement plans and future lines of work.

Keywords: sensors, sensor network, wireless communication, cultivation, monitoring cultivation.

Índice

Capítulo 1. Introducción

Desde su inicio en el período Neolítico, cuando la economía de las sociedades humanas evolucionó de la recolección, la caza y la pesca a la agricultura y la ganadería, hasta el día de hoy, la agricultura ha sido, es y se espera que sea, una de las principales actividades económicas basada en la explotación de los recursos que la tierra origina y que ha permitido desde los tiempos más remotos la supervivencia, manutención e, incluso, la remuneración principal de muchas familias en todas las partes del mundo.

Entendemos por agricultura el conjunto de técnicas y conocimientos necesarios para cultivar la tierra, siendo pues una actividad de lo que actualmente se conoce como sector primario. En ella se engloban los diferentes trabajos de tratamiento del suelo y los cultivos de vegetales, tales como cereales, frutas, hortalizas, pastos cultivados y forrajes, comprendiendo todo un conjunto de acciones humanas (y a veces también animales, en el caso del arado de tiro animal) que transforma el medio ambiente natural.

Como bien se ha comentado anteriormente, la agricultura nació en el Neolítico por la zona del Oeste de Asia, Egipto y la India, que sembraron por primera vez plantas (trigo y cebada) que previamente habían sido recogidas de la naturaleza. Pero donde realmente se desarrolló de manera muy favorable el cultivo fue en el Norte y Sur de China, África y Nueva Guinea (y alguna región de América). Pasando por la Antigua Roma, donde el principal cultivo eran cereales, leguminosas, hortalizas y trigo, la Edad Media y la introducción de innovaciones como el arado y nuevos tipos de hoces y guadañas, así como la rotación trienal para sanar las tierras, la Edad Moderna y su Revolución Agrícola previa a la Revolución Industrial, hasta la Edad Contemporánea (Actualidad) donde se usan abonos químicos y se ha introducido muy fuertemente la mecanización y la biotecnología en nuestros campos, comprobamos que dicha actividad ha sido decisiva y fundamental en muchos de los momentos importantes de la historia de la humanidad y su aparición y desarrollo ha sido muy necesaria para la evolución del ser humano.

Pero, a pesar de que actualmente la actividad agrícola se sigue realizando, hay que tener en cuenta que a partir de la segunda mitad del siglo XX ésta entró en decadencia. La gran inversión del siglo anterior en agricultura debido a los altos ingresos que se obtenían, que produjo la Industrialización Agrícola y su posterior deterioro, y la aparición de un nuevo concepto conocido como Éxodo Rural (emigración del campo a la ciudad) fueron los puntos clave para el declive de la agricultura. Éste último provocó, principalmente, que la actividad agraria envejeciera, debido a la migración de los jóvenes a las ciudades en busca de nuevas oportunidades.

Actualmente, se están buscando soluciones efectivas para evitar que, a pesar del envejecimiento del personal agrícola, se pueda tener el campo en perfecto estado y controlado durante todo el día sin estar presente en él y evitar así la desaparición de la actividad. Hoy en día, en muchas familias son los padres/madres (abuelos/abuelas) los que se encargan del cultivo de tierras para su propia manutención, venta de productos, o incluso trabajan para grandes compañías agrícolas. La presencia juvenil en esta actividad ha pasado de abundante a casi inexistente. En contraposición, la evolución de la tecnología es un camino favorable para el cuidado y control de cultivos, que puede ser implantada en situaciones donde ningún familiar (dígase por enfermedad, indisponibilidad, trabajo en ciudades u otras inoportunidades) pueda trabajar/controlar el cultivo en un momento temporal o, incluso, solo pueda controlarlo presencialmente en momentos muy específicos, sin estar presente en el mismo. No se pretende que se tome este camino como un intento de mecanizarlo todo como excusa para automatizar las tareas del campo y así evitar el personal agrícola necesario, si no que se vea a la tecnología como una “amiga” que nos puede servir en situaciones extremas, o no tanto, para monitorizar

todos los parámetros que le pudieran afectar a un cultivo, tales como son la temperatura a la que está expuesto, la humedad, el color de la hoja, el color del fruto, etc., y, si fuese necesario, poder domotizarlo todo de tal manera que, si por un casual no se pudiera acceder al cultivo durante un periodo de tiempo lo suficientemente largo como para pensar que podría estar expuesto dicho cultivo, se pudiera realizar desde casa toda tarea relacionada con el cuidado del cultivo, como regar, usar abonos y fertilizantes, aumentar/disminuir la temperatura, desplegar coberturas artificiales anti-luz, etc.

Claro está que el cuidado manual/presencial del campo se puede considerar más efectivo debido al temor de muchos agricultores (sobre todo el sector más envejecido) al fallo de la tecnología; ya que, si se piensa, un fallo en la detección de humedad (u otro parámetro) puede provocar un deterioro catastrófico e irreparable de nuestros cultivos. Sin embargo, en este presente, se mostrará una solución sencilla, tecnológica, moderna y eficaz de monitorización de los parámetros más cercanos a los cultivos, específicamente en el cultivo del tomate. Además, todos estos datos acerca de la situación del cultivo se tendrán a tiempo real en nuestro Smartphone o Tablet para que, en caso de que no se pueda acceder a dicho cultivo por movilidad (u otras), se disponga de un medio de comunicación que nos desglosará la situación del mismo y las posibles medidas a tomar. También, por último, se hablará de la toma de decisiones y la visión de futuro en cuanto a domótica se trata. Una vez introducido el concepto de agricultura, así como la funcionalidad del trabajo que desde ahora se presenta, se procede a realizar una síntesis sencilla de los objetivos a alcanzar. Puesto que en este trabajo hay que aplicar conocimientos adquiridos de la ingeniería de Telecomunicaciones, se propondrá un sistema que combinará electrónica analógica y digital, bases de datos, programación en Android y Arduino y otros conceptos que veremos más adelante, todo ello para poder favorecer el crecimiento de la tomatera que tomaremos como ejemplo en cuestión.

Capítulo 2. Objetivos

- Entender cuáles son las magnitudes físicas principales que afectan a los cultivos

- Comprender cómo afectan dichas magnitudes a los cultivos

- Elegir, de forma lógica, los sensores más óptimos que nos permitan cuantificar dichas magnitudes físicas

- Almacenar todos los datos recogidos desde el momento que se conecte el sistema para tener información actual de nuestro cultivo.

- Acceder a la información del estado del campo a tiempo real mediante nuestro Smartphone o Tablet.

- Hablar acerca de las futuras propuestas de trabajo que puede plantear el presente proyecto, comentando las posibilidades de mejora que pudiera tener y la aplicabilidad del sistema.

Capítulo 3. Metodología

Para poder entender la metodología que se ha seguido a la hora de realizar este proyecto, hay que hablar de tres partes fundamentales: la gestión del proyecto, la distribución de tareas y el diagrama temporal utilizado; tres conceptos que han permitido llevar el trabajo de una forma más organizada, fijando unos objetivos y unos tiempos de realización claros.

3.1 Gestión del proyecto

Entendemos la gestión de un proyecto como la disciplina del planeamiento, la organización, la motivación y el control de los recursos con el propósito de alcanzar los objetivos impuestos inicialmente. En resumen, es un enfoque metódico para planificar y orientar los procesos del proyecto de principio a fin. Para esta memoria, se ha realizado un organigrama personal para explicar de forma más visual la gestión del proyecto mediante las actividades que se han de realizar para conseguir los objetivos de la mejor manera y controlando cada tarea con su debido seguimiento, siendo el resultado el presentado en la Figura 1:

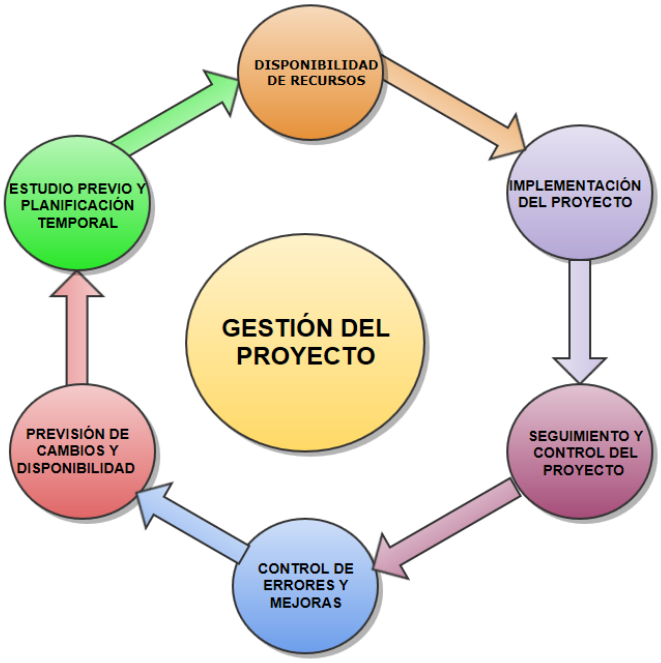


Figura 1. Organigrama de la gestión del proyecto

Como explicación breve, se aclarará explícitamente a qué corresponde cada fase del organigrama de la Figura 1 con el proyecto en cuestión. En primer lugar, tomaremos como fase inicial los estudios previos y la planificación temporal, esto es, entender qué es lo que queremos, qué es lo que buscamos, qué es lo que hay que saber para realizarlo y cuánto tiempo tenemos para terminarlo y así poder planificarlo temporalmente. Seguida de esta fase, se encuentra la disponibilidad de recursos, dicho de otra manera, qué necesitamos para poder construir nuestro proyecto (en este caso, sensores, cables, placa Arduino, etc.) y, si no

disponemos de ello, planificar su compra y el presupuesto. A continuación se realizaría la etapa principal, la implementación del proyecto. Esto abarca desde el montaje, las primeras líneas de código, etc. hasta la última actividad que se haga en lo que se refiere a implementación. En paralelo, aunque parezca que va a continuación, estaría el control de errores y mejoras. Para una mejora en la eficacia y en tiempo, desde el primer momento se han ido haciendo testeos unitarios y del sistema, para evitar errores en las últimas fases del proyecto que serían difíciles de localizar/solventar y, así, detectarlos en fases tempranas que nos permitirá solucionarlos rápidamente y no provocar así retrasos innecesarios. Por último, una fase de previsión de cambios y disponibilidad para poder cambiarlos. De lo que trata es de que, si se encuentra un error o se necesita meter un cambio en el proyecto, se pueda solucionar con la mayor rapidez y eficacia posible. Se conecta con la fase de estudio previo y planificación temporal debido a que, en caso de error o cambio, se necesita recalcular la previsión de tiempos e, incluso, hacer nuevos estudios previos. Finalmente, una perfecta ejecución de estas 6 fases darán como resultado una gestión del proyecto impecable y eficiente.

3.2 Distribución de tareas

Para este proyecto, la distribución de tareas es algo más sencilla que para un proyecto grupal típico en empresas de telecomunicaciones actuales. Al ser una única persona el que ha realizado todo el proyecto, toda subtarea que se desglosara de la tarea principal será llevaba a cabo por la misma persona. Por lo que, en lo que consta a distribución de tareas se ha seguido un patrón unilateral, lo que viene a significar que, en caso de que salga una nueva subtarea, es el ejecutador principal (el realizador del trabajo) quien ha de conseguir realizarla. Aún así, se ha decidido dividir el proyecto general en tareas y subtareas para llevar un mejor control de todo lo realizado y poder así gestionar de una manera más eficaz los recursos y el tiempo disponible. La división de estas tareas se muestran en el siguiente organigrama de la Figura 2:

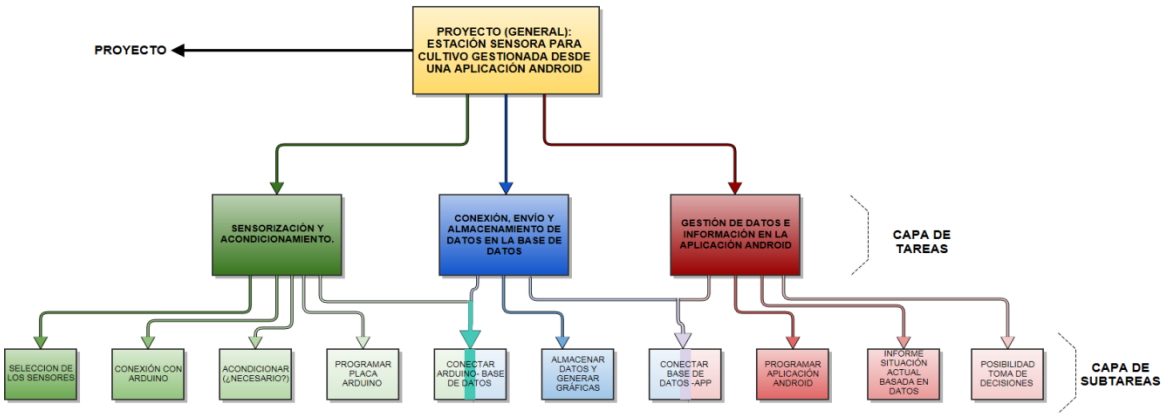
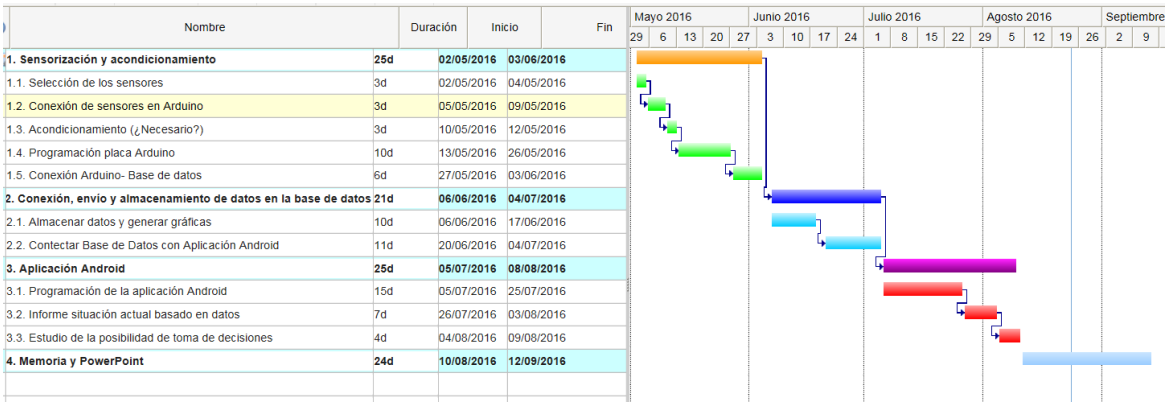


Figura 2. Organigrama de la división del proyecto en tareas y en subtareas

Como se puede comprobar, hay tres tareas principales: la primera trata sobre la elección de los sensores bajo criterio, conectarlos, acondicionarlos si lo necesitan (digitales normalmente no, analógicos sí), programar todo el Arduino y conectarlo a la base de datos a la que enviaremos toda la información pertinente. La segunda tarea se resume en la conexión del Arduino con la base de datos, la recepción de dichos datos y el almacenamiento de los mismos para que luego puedan ser exportados en caso de necesitarlos. Por último, la gestión de datos y toma de decisiones en la aplicación de Android, lo que consiste en conectar con la base de datos desde la app que ya estará previamente programada, visualizaremos datos de interés y, la propia aplicación, nos dará recomendaciones con las posibles acciones/medidas que deberíamos tomar. A todo esto hay que sumarle una tarea pre-desarrollo del sistema completo: la búsqueda de información y documentación sobre los cultivos, los parámetros que les afectan, etc. El conjunto de todo esto conforma el proyecto actual, la estación sensora.

3.3 Diagrama temporal

Finalmente, para acabar con la metodología aplicada para realizar este proyecto, se hizo al principio un diagrama temporal de Gantt, tal como se muestra en la Figura 3, en el cual se puede observar cual era la primera impresión temporal que se tuvo acerca de la realización del proyecto. Se puede observar que se añadió una cuarta etapa temporal, que se había reservado previamente para poder realizar la memoria del trabajo y el PowerPoint necesarios para poder presentarlo:



Nombre	Duración	Inicio	Fin
1. Sensorización y acondicionamiento	**25d**	**02/05/2016**	**03/06/2016**
1.1. Selección de los sensores	3d	02/05/2016	04/05/2016
1.2. Conexión de sensores en Arduino	3d	05/05/2016	09/05/2016
1.3. Acondicionamiento (¿Necesario?)	3d	10/05/2016	12/05/2016
1.4. Programación placa Arduino	10d	13/05/2016	26/05/2016
1.5. Conexión Arduino- Base de datos	6d	27/05/2016	03/06/2016
2. Conexión, envío y almacenamiento de datos en la base de datos	**21d**	**06/06/2016**	**04/07/2016**
2.1. Almacenar datos y generar gráficas	10d	06/06/2016	17/06/2016
2.2. Contectar Base de Datos con Aplicación Android	11d	20/06/2016	04/07/2016
3. Aplicación Android	**25d**	**05/07/2016**	**08/08/2016**
3.1. Programación de la aplicación Android	15d	05/07/2016	25/07/2016
3.2. Informe situación actual basado en datos	7d	26/07/2016	03/08/2016
3.3. Estudio de la posibilidad de toma de decisiones	4d	04/08/2016	09/08/2016
4. Memoria y PowerPoint	**24d**	**10/08/2016**	**12/09/2016**

Figura 3. Diagrama temporal del proyecto

Como se puede comprobar, la duración total del proyecto se calculó para poder realizarse en 100 días. Podemos ver que las tareas se reparten más o menos equitativamente (de media unos 22.5 días por tarea principal), aunque vemos que el tiempo dedicado a cada subtarea ya es más desigual, dándole más importancia a las tareas más relevantes. Hay que considerar que este diagrama se ajusta a la fase de desarrollo, no se ha tenido en cuenta la fase previa de documentación y búsqueda de información, la cual tuvo una duración de 1 mes, empezando el proyecto realmente el lunes 4 de abril y añadiendo al diagrama 28 días inicialmente.

Más adelante veremos si finalmente se cumplió estos objetivos tal cual se expone en la Figura 3.

Capítulo 4. Parámetros a medir

Como parte de los estudios previos a realizar para poder empezar a trabajar en la implementación del proyecto, y una vez que se tiene los objetivos fijados y se entiende claramente la metodología a emplear y el tiempo disponible, se procede a analizar las magnitudes físicas (parámetros) que puedan influir en nuestro cultivo. Esto es, se va a desarrollar una lista de parámetros que deberemos cuantificar, ya que estos afectan notablemente a nuestro cultivo, por lo que tendrán que estar monitorizados en todo momento para que, en caso de que las medidas se salgan de los valores normales/esperados, poder actuar a tiempo y proteger así nuestro cultivo de condiciones no favorables para su crecimiento.

Por resumir brevemente, la lista de magnitudes físicas a medir es la siguiente: humedad relativa del aire, temperatura, intensidad de luz recibida (tanto visible como infrarroja), el color de las hojas (o fruto), la humedad del terreno, la presión atmosférica y, por último, otras medidas de interés (CO_2, contaminación del aire, etc.) que se desarrollarán en el apartado oportuno.

4.1 Humedad relativa del aire

La humedad del aire se define como la cantidad de vapor de agua presente en el aire. Dicha humedad se debe al vapor de agua procedente de la evaporación de mares, océanos, ríos, etc., que se encuentra presente en la atmósfera. La cantidad de vapor de agua que puede absorber el aire depende de la temperatura a la que se encuentre el ambiente. Por especificar un poco más, si el aire es caliente, éste admite más vapor de agua. Por el contrario, a bajas temperaturas, el aire admite menos vapor de agua.

El concepto de humedad relativa viene de definir dicha humedad como la relación entre la fracción molar del valor de agua en el aire y la fracción molar del vapor de agua en el aire saturado a la misma temperatura, tal como se puede observar en la ecuación (4.1):

$$\phi = \frac{P_v}{P_{vs}} * 100 \qquad \textbf{(4.1)}$$

Cuando se habla de cultivos, la humedad es un elemento clave para que estos florezcan o, sin más remedio, se echen a perder. En el caso de los invernaderos, para situarnos y hacernos una idea, la humedad puede ser el favor ambiental más incontrolable ya que, en ocasiones, ni los mejores equipos de control ambiental pueden controlar dicho parámetro a la perfección. Hay que tener en cuenta que los niveles de humedad, como bien se han dicho anteriormente, dependen de la temperatura, es decir, fluctúan con el cambio de la temperatura del aire. Si a eso se le suma que las plantas transpiran y agregan vapor de agua al ambiente, tenemos una variable un tanto inestable en ocasiones.

El cómo puede afectar el aire demasiado húmedo a nuestro cultivo se resume en enfermedades de las raíces y las hojas, secado lento del sustrato, estrés de las plantas, pérdida de calidad del producto y alta probabilidad de pérdida de la producción. Por el contrario, con bajo nivel de humedad, el crecimiento de las plantas se puede ver comprometido, haciendo que los cultivos tarden más tiempo en obtener un tamaño apto para consumo/venta. Las plantas, para intentar adaptarse a la humedad, ajustan las aberturas de las estomas de las hojas, aunque en ocasiones con humedades extremas (las citadas anteriormente, muy altas o muy bajas) no logran adaptarse del todo.

Podemos observar un resumen de cómo afecta la humedad a las plantas en la Tabla 1:

Humedad demasiado baja	Humedad demasiado alta
Marchitamiento	Crecimiento débil
Plantas atrofiadas	Aumento de enfermedades de las hojas
Tamaño más pequeño de las hojas	Deficiencias de nutrientes
Puntas secas y quemadas	Aumento de enfermedades de las raíces
Hojas rizadas	Edemas
Aumento de la infestación de arañuela roja	Bordes quemados (gutación)

Tabla 1. Problemática en plantas con condiciones de humedad fuera de los valores normales

Es por todo esto que se necesitan tener las mediciones pertinentes de humedad y poder contar con ellas en cualquier momento para que, en el caso de que los valores de la humedad excedan los límites de la normalidad para el crecimiento del cultivo, éste no se vea comprometido y se actúe con rapidez, volviendo a establecer los valores de temperatura óptimos, ya sea con humidificadores o acondicionando el aire.

Teniendo en cuenta que el estudio de nuestro cultivo se va a realizar en un pueblo de Albacete, Ossa de Montiel, podemos realizar una consulta de los datos históricos de humedad relativa por esa zona, los cuales se han graficado en la Figura 4:

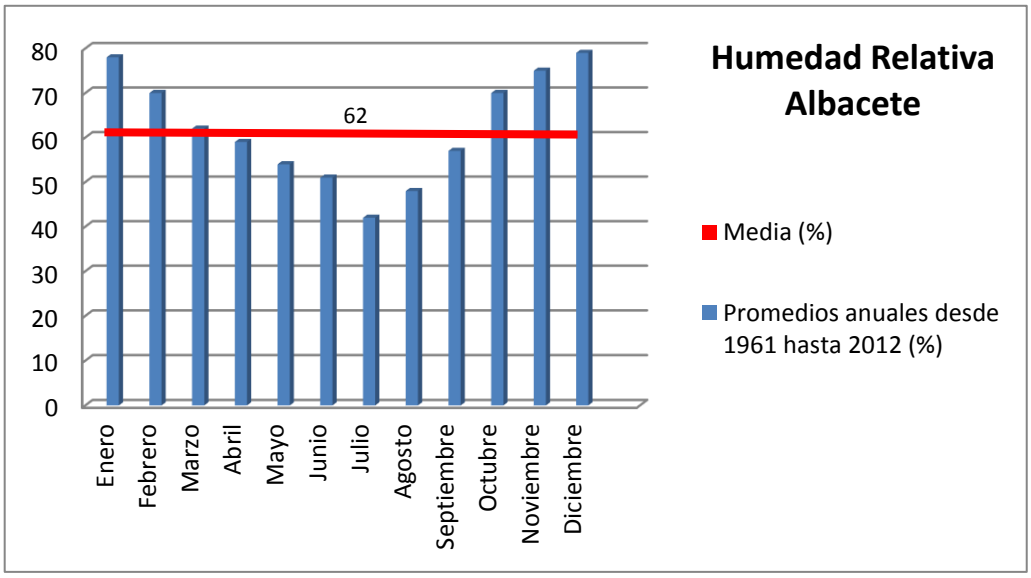


Figura 4. Gráfico-resumen de los promedios de humedad relativa en Albacete desde 1961 hasta el 2012

Hay que tener en cuenta que los valores de humedad "normales" dependerán del tipo de cultivo a analizar, ya que éstos van a variar entre diferentes tipos de plantas; como bien se dijo en la introducción, nuestra planta a controlar será una tomatera. Para el caso de los tomates, la humedad relativa óptima oscila entre el 65 – 70%. Dentro de este rango se favorece el desarrollo normal de la polinización, garantizando así una buena producción. En condiciones de baja humedad relativa, la tomatera sufrirá estrés hídrico, aumentando el consumo de agua y eliminando la consumición de nutrientes, esto es, una disminución de su crecimiento y un cumulo de sales en el medio. Por el contrario, con valores extremadamente altos se reducirá la absorción de agua y nutrientes y se favorecerá la disminución de elementos como el calcio, provocando desórdenes fisiológicos. Debemos evitar, pues, humedades por debajo del 45% y por encima del 85%. En el caso de Albacete, se deberá tener mayor vigilancia en los meses de enero, noviembre y diciembre (alta humedad), y los meses de julio y agosto (baja humedad).

4.2 Temperatura

La temperatura es la magnitud física más común a tener en cuenta a la hora de realizar un cultivo. Como definición podremos decir que es una magnitud física que refleja la cantidad de calor, ya sea de un cuerpo, de un objeto o del ambiente en general (para este trabajo, la temperatura a tener en cuenta será la del ambiente). Cuando se habla de ésta, hay que mencionar lo que se llaman nociones de temperatura. Las nociones de temperatura son las sensaciones que provoca la misma, es decir, noción de frío, debida a bajas temperaturas, y noción de calor, debida a altas temperaturas.

Sí hablamos de la temperatura desde el punto de vista físico, hay que saber que está relacionada con la energía interior de los sistemas termodinámicos, basándose en el concepto de movimiento de las partículas que confortan dicho sistema, y que cuantifica la actividad de las moléculas de la materia, lo que viene a decir que a mayor energía sensible, mayor temperatura. O dicho de otro modo, a mayor velocidad de partículas, mayor calor desprendido.

Hay que tener en cuenta que el estado, la solubilidad y el volumen de la materia dependen directamente de la temperatura. Esto se puede explicar fácilmente con el ejemplo del agua expuesta a diferentes temperaturas en la Figura 5:

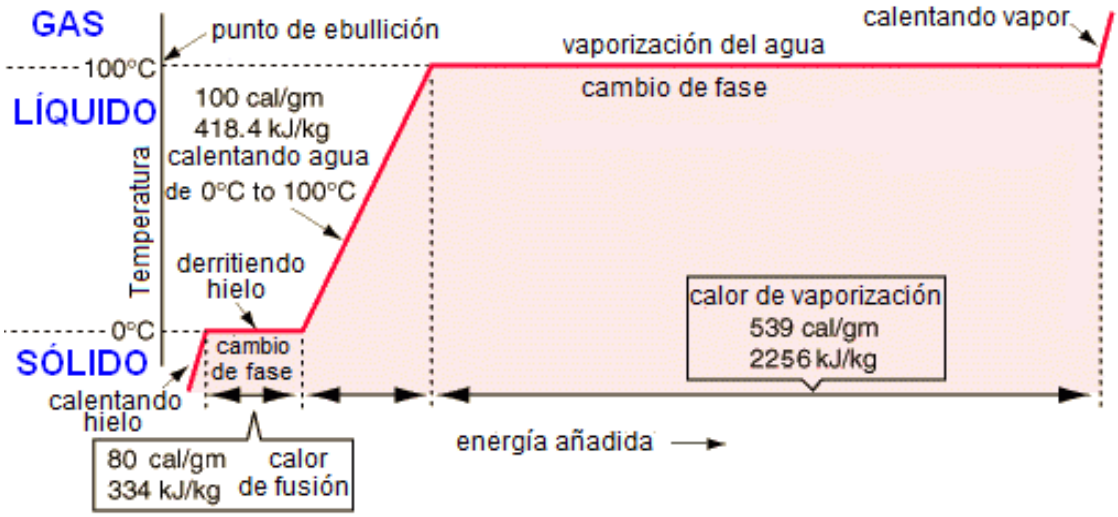


Figura 5. Estados del agua dependiendo de la temperatura

Como se puede observar, a una temperatura menor de 0ºC estaremos en el estado sólido del agua, donde ésta se transforma en hielo. Si aumentamos la temperatura, estableciendo un valor entre 1ºC y 99ºC, el agua se transforma en líquido. Por último, si aumentamos aún más la temperatura, pasando los 100ºC, entraremos en el estado gaseoso (vapor). Si tomamos la definición física de temperatura anteriormente citada, es decir, el concepto de movimiento de partículas, concluimos en que en el estado sólido las partículas apenas se mueven; en el líquido, se mueve más deprisa que en el sólido pero menos que en el gaseoso, donde las partículas se mueven muy rápidamente.

Se procede, pues, a explicar porque es necesario tener esta medida en cuenta a la hora de cultivar el producto que, de nuevo, será una tomatera. La temperatura del aire es la principal magnitud a medir en el ambiente ya que es el parámetro que más influye en el desarrollo vegetativo, desarrollo de racimos florales, la producción y crecimiento del fruto, así como su maduración y calidad del mismo.

Al igual que con el caso de la humedad, se procede a graficar (en la Figura 6.) los datos históricos de temperatura de la zona de Albacete, lugar donde se encuentra nuestro cultivo:

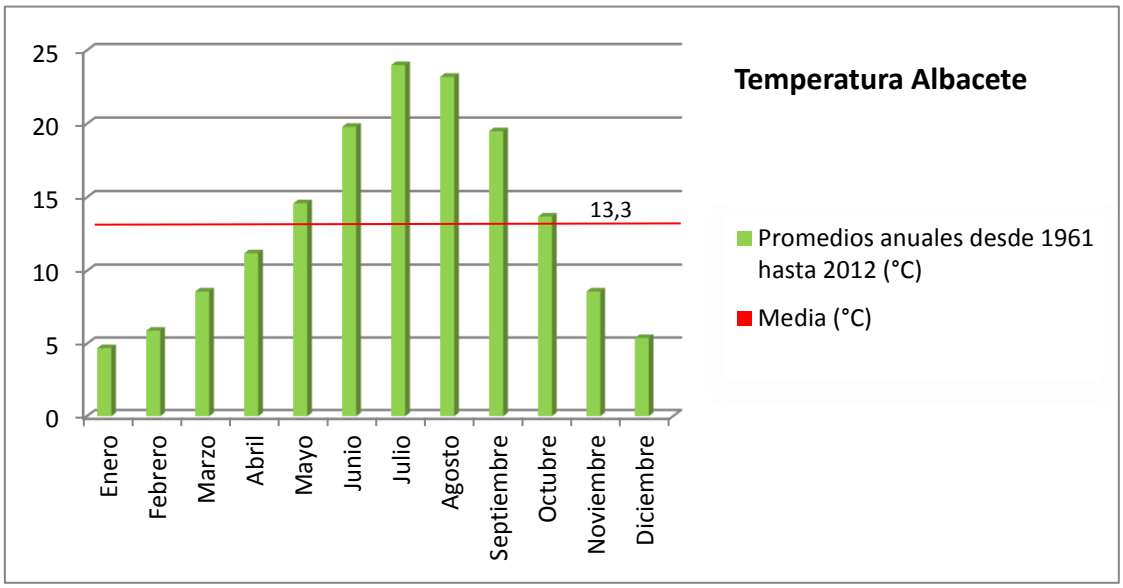


Figura 6. Gráfico-resumen de los promedios de temperatura en Albacete desde 1961 hasta el 2012

Los rangos de temperatura para un óptimo desarrollo del cultivo varían entre los 18-30ºC durante el día y 10-25ºC durante la noche. Temperaturas por encima de los 30º o por debajo de los 10º se consideran críticas durante la floración, provocando la caída de la flor y limitando el desarrollo del fruto. Es por todo esto que la temperatura es un parámetro muy crítico para el crecimiento de nuestros tomate, por lo que es necesario monitorizar durante todo el día la temperatura y, en caso de que se excedan los límites que hayamos impuesto dependiendo de la variedad del tomate, avisar al agricultor de dichas temperaturas y que éste reaccione, por ejemplo, activando el sistema de ventilación climático y regule dicha temperatura, salvando así la pérdida de su género. En el caso de Albacete, vemos que enero, febrero, marzo, noviembre y diciembre son meses críticos a tener muy en cuenta.

4.3 Intensidad de luz recibida

Se define intensidad luminosa recibida como la cantidad de flujo luminoso que es capaz de absorber un cuerpo material. Es la cantidad absorbida de flujo luminoso que emite una fuente por unidad de ángulo sólido. En el caso que nos atañe, las fuentes lumínicas a tener en cuenta serán tanto naturales (Sol) como artificiales (lámparas/fotoemisores de luz en invernaderos).

Nuestros cultivos son capaces de percibir tres tipos de rangos lumínicos: la luz visible, los rayos infrarrojos y la luz ultravioleta. Esta última no se tendrá en cuenta en este proyecto debido a que no afecta en una gran proporción al cultivo, siendo las otras dos las más críticas y, por ello, motivo fundamental de estudio.

4.3.1 Espectro Visible

Se considera espectro visible a la región del espectro electromagnético que el ojo humano es capaz de percibir. A la radiación electromagnética que compone este rango de longitudes de onda se le conoce como luz visible, cuyo espectro junto a las longitudes de ondas correspondientes se puede observar en la Figura 7. A pesar de que un típico ojo humano suele responder a longitudes de onda desde 390 nanómetros (nm) hasta 750 nm, existen personas que pueden percibir longitudes desde 380nm hasta 780 nm. Por ejemplificar, el arcoíris sería un perfecto dechado de refracción de espectro visible.

Color	Frecuencia	Longitud de onda
Violeta	668–789 THz	380–450 nm
Azul	631–668 THz	450–475 nm
Ciano	606–630 THz	476–495 nm
Verde	526–606 THz	495–570 nm
Amarillo	508–526 THz	570–590 nm
Naranja	484–508 THz	590–620 nm
Rojo	400–484 THz	620–750 nm

Figura 7. Espectro de luz visible

Para nuestro cultivo, la luz solar (artificial en su defecto) es un pre-requisito necesario para el desarrollo de nuestro cultivo. Si hacemos un análisis desde el punto de vista de la biología, el crecimiento de una planta es producido por un proceso conocido como fotosíntesis, esto es a grandes rasgos, una transformación de sustancia inorgánica en orgánica realizada por células vegetales gracias a la transformación de la energía luminosa en la química producida por la clorofila (durante la fotosíntesis, las plantas absorben dióxido de carbono y expulsan oxígeno). La fotosíntesis, pues, sólo se da cuando la luz es absorbida por la clorofila en las partes verdes de la planta, localizadas principalmente en las hojas de la misma.

Para el caso del tomate, el tiempo de exposición a la luz visible no es algo crítico, ya que no le afecta el fotoperiodo debido a que sus necesidades lumínicas oscilan entre 8 y 16 horas; eso sí, requiere una buena iluminación durante esas horas. Los días soleados y sin ninguna interferencia (como las nubes), el crecimiento y el desarrollo normal del cultivo se encuentra estimulado. Es conocido que Castilla-La Mancha es una región bastante soleada durante todo el año, por lo que problemas de iluminación no se esperan tener. Pero, por si acaso, se procederá a la implementación de un sistema de medida del nivel de la misma, asegurándonos que éste no esté por encima de una nivel estimado; si no fuera así, se notificará al agricultor de dicha situación para que tome medidas pertinentes. Hay que percatarse de que en verano el sol tiene una trayectoria más alta en el cielo y la radiación solar incide con mayor intensidad, siendo pues la intensidad de la luz mayor en verano que en invierno.

Hay que tener en cuenta también que la distancia entre una mata de tomates y otra es crucial para el desarrollo de las primeras flores, debido a que se puede dar el caso de poca luz al taparse

una con la otra, por lo que a la hora de medir será muy importante colocar el sensor en una zona donde capte sin ningún tipo de problema la luz visible directa.

4.3.2 Radiación Infrarroja

Gracias a la aparición de los termómetros infrarrojos, los cuales permiten cuantificar la cantidad de infrarrojos cercano (800 nm ~ 2500 nm) que nos rodean, fuimos capaces de observar como dicha radiación nos rodeaba y en qué medida. Aunque la radiación Ultravioleta se quede un poco al margen por ser fácilmente eliminada en los cultivos con protectores, hay que tener especialmente cuidado con la infrarroja.

Para hacer una corta introducción de cómo afectan a nuestros cultivos, se hace saber que las plantas consisten en su mayor parte de agua, por lo que se calientan muy rápido con la presencia de altas cantidades de radiación infrarroja. Todo esto se debe a que el agua absorbe el infrarrojo muy bien, por lo que al ser absorbido por una planta, la temperatura de la misma aumenta, provocando situaciones catastróficas para nuestro cultivo como la pérdida del producto, malformaciones, etc. Como conclusión podemos determinar que los infrarrojos no provocan ningún beneficio en la planta, por lo que lo óptimo sería bloquearla, por ejemplo, con pantallas.

En cantidades controladas, dentro de cierto rango, es positivo que el cultivo se caliente. Pero hay límites, ya que si combinas una gran cantidad de luz visible, altas temperaturas en las plantas debido a infrarrojos y una baja humedad producirían un parón en la fotosíntesis, dañando así el cultivo. Por todo esto es por lo que es ve necesario medir para el caso de nuestra tomatera la radiación infrarroja existente; para que, en el caso de que se acumule grandes cantidades de infrarrojos, notificar al agricultor y que tome las decisiones necesarias (apantallando u otras soluciones), evitando así una catástrofe irreparable en el cultivo.

4.4 Color de la hoja

El control sobre el color de la hoja para cualquier planta es importante, pero en el caso de la tomatera, la cual genera un producto consumible, el color de la hoja es fundamental. Éste dice mucho sobre el estado de salud de la planta. Como es obvio, detectar a tiempo una enfermedad del cultivo hace que, aplicando los medios necesarios a cada caso, se pueda salvar dicho producto.

Existen varias enfermedades destacables que se pueden detectar a partir del color de la hoja. Siempre tenemos que tener como referencia el "verde hoja tomate" normal de una tomatera en pleno estado de salud, mostrado en la Figura 8 junto a su correspondencia con los valores RGB en la Tabla 2 que más tarde utilizaremos en la implementación del sistema:

Figura 8. Hoja sana tomatera

RGB

R	076
G	145
B	065

Tabla 2. Correspondencia RGB hoja tomatera

Partiendo de esos datos de referencia, se procede a analizar brevemente los 3 tipos de enfermedades habituales de las hojas. En primer lugar tenemos Oidio (Figura 9), una enfermedad provocada por anomalías en las condiciones de temperatura y humedad, provocando la decoloración de la hoja hacía un color blanquecino, que puede llegar a tornar amarillo. La segunda enfermedad es Mildiu (Figura 10), un hongo que provoca el aclarado de las hojas de las plantas en forma de mancha. Por último, Roya (Figura 11), un hongo que vuelve la hoja de color naranja en el envés y amarilla en el haz. Aunque el fin de este trabajo no será analizar dichas enfermedades, si habrá que tenerlas en cuenta para futuras decisiones y control de enfermedades.

Figura 9. Oidio en la hoja de la tomatera

Figura 10. Mildiu en la hoja de la tomatera

Figura 11. Roya en la hoja de la tomatera

Además de conseguir interceptar la enfermedad a tiempo para que no eche a perder nuestro producto, también es interesante observar el color de la hoja para poder determinar si existe, o no, deficiencia de nutrientes, tales como hierro, calcio, magnesio, fosfatos, etc., en nuestro cultivo. Para ello, como resumen, se puede observar que le ocurre a la hoja de la planta en los casos anteriores visualizando la Figura 12, donde se explica cuáles son las repercusiones en la hoja cuando hay déficit de nutrientes:

Figura 12. Anomalías en las hojas por falta de nutrientes.

Como se puede comprobar, todo lo anterior repercute en malformaciones y decoloraciones en la hoja por lo que, monitorizándolo adecuadamente, se podría llegar a predecir la enfermedad o los déficits de nutrientes mucho antes de que ya sea irreversible, ayudando así a la buena producción del cultivo. También, si se quiere, se puede hacer un seguimiento del fruto, desde su inmadurez (verde) hasta el producto final (rojo) para ir comprobando que todo va correctamente a lo largo del periodo.

4.5 Humedad del terreno

Se entiende por humedad del terreno como la cantidad de agua por volumen de tierra que hay en el suelo y éste debe ser capaz de mantener.

Durante el crecimiento del cultivo, la tomatera necesita mantener la humedad del suelo, por lo que la cantidad de veces que debemos regar dependerá de en qué estado se encuentre el terreno donde esté la planta. En el supuesto caso de tener un suelo muy suelto, se deberá regar durante todos los días, puede ser que hasta incluso un par de veces al día, intentando regar durante las horas de menos calor (madrugada y tarde/noche) para evitar así la rápida evaporación del líquido. Por otro lado, con terrenos más compactos no hace falta regar día tras día, pero si hace falta comprobar que dicho suelo se encuentre húmedo para poder aportar agua cuando el suelo más profundo se encuentre seco. Es por ello que vamos a monitorizar también la magnitud

física de la humedad del terreno, para evitar la sequía del suelo y, si ésta se produjera, el sistema avisará rápidamente al agricultor para que éste proceda al riego de la planta. Obvio es que, si exponemos la tomatera a largos periodos de sequedad del terreno, ésta morirá desnutrida al no poder aportarle agua con minerales a su sistema.

4.6 Presión atmosférica

La presión atmosférica es la fuerza por unidad de área que ejerce el aire sobre la superficie terrestre. En este caso, la presión atmosférica no es un parámetro que realmente afecte a las plantas directamente, es decir, variaciones de la presión atmosférica estable no producen anomalías directas en la planta como las anteriores; pero esta magnitud física si puede servir para realizar pronósticos del tiempo. Esto es, teniendo tantos datos como sea posible acerca del estado de la atmósfera, tales como temperatura, presión atmosférica y humedad, y aplicando patrones y lógicas matemáticas, se puede llegar a predecir (no es segura al 100%, pero puede servir de guía) el tiempo en un determinado momento futuro.

La utilidad de esta magnitud en nuestro cultivo será la de, como ya hemos comentado antes, predicción del tiempo atmosférico. La tomatera requiere un suelo que no esté seco, pero tampoco se necesita un suelo encharcado el cual provoque la muerte de la planta. Se necesita un suelo húmedo, que se consigue regulando el agua aplicada a los tomates. Ahora bien, se ha de tener muy en cuenta la probabilidad de lluvia para que, si lloviera, no accionar el regado. Si se ha regado y, más tarde, se pone a llover provocaremos el ahogamiento de la planta. Es por ello que, una predicción a tiempo, puede ayudarnos mucho a mejorar el crecimiento de nuestra planta, así que, por el bien de nuestro cultivo, se tendrá que comprobar siempre la presión que, junto a la temperatura y a la humedad, nos indicará si hay posibilidad de lluvia.

Para entender un poco como funciona, se va a explicar en qué se va a basar la predicción del tiempo; en nuestro caso utilizaremos dos medidas potentes que teniéndolas en cuenta a la vez se puede llegar a predecir el tiempo con un alto nivel de precisión: la temperatura y la humedad. Teniendo en cuenta para ello que la temperatura normal media es de unos 18º y la presión atmosférica media es de 1013 milibares, nos basaremos en la Tabla 3 para realizar las predicciones pertinentes:

PRESIÓN ATMOSFÉRICA	TEMPERATURA	PREDICCIÓN
BAJANDO	BAJANDO	LLUVIA ABUNDANTE
BAJANDO	ESTABLE	PROBABILIDAD LLUVIA
BAJANDO	SUBIENDO	TIEMPO VARIABLE
ESTABLE	BAJANDO	PROBABILIDAD LLUVIA
ESTABLE	ESTABLE	TIEMPO VARIABLE
ESTABLE	SUBIENDO	BUEN TIEMPO
SUBIENDO	BAJANDO	VIENTO POLAR
SUBIENDO	ESTABLE	BUEN TIEMPO
SUBIENDO	SUBIENDO	TIEMPO CALUROSO Y SECO

Tabla 3. Predicciones basadas en la presión atmosférica junto a la temperatura

En los casos de predicción de lluvia abundante no es recomendable regar. En el caso de probabilidad de lluvia, habría que hacer un estudio más exacto acerca de esa posibilidad de

lluvia; aunque no se aconseja, se podría regar. Si hay tiempo variable, se deja a la decisión del agricultor, ya que no es posible predecir dicho tiempo con el sistema que tenemos. Si hace "buen tiempo" o, incluso, caluroso y seco, se aconseja proceder al regado del terreno.

Como normal general, se miraran también los datos del sensor de humedad del terreno debido a que esta predicción atmosférica podría fallar. Si nos sale que el suelo está ya húmedo, ya sea por lluvias o regados previos acumulados, no deberemos regar, nos dé la predicción que nos dé.

4.7 Otras medidas de interés

Aunque en este trabajo no se van a implementar, la monitorización/control de los siguientes parámetros puede ser de especial interés para los agricultores en el caso que quieran tener una plantación controlada al 100% de todo riesgo existente. Entre otras, se ha decidido destacar las que a juicio propio son más importantes y podrían perjudicar mucho a la planta tomatera.

En primer lugar, un parámetro que se debería controlar desde el primer momento es la calidad del agua. Una mala calidad del agua podría provocar que el producto fuera no apto para el consumidor, generando enfermedades para el mismo, y echar a perder todo el cultivo. Es por ello que, si se tuviera ocasión, un control de la calidad del agua sería necesario para asegurar la calidad de nuestro producto.

Seguidamente, en segunda posición, se debería controlar el pH de la tierra. El pH es un parámetro que indica el grado de acidez o alcalinidad de un medio acuoso. No tiene importancia directa en el desarrollo de la planta, pero si indirecta, radicando su influencia en la presencia de iones tóxicos, estando el valor de pH normal entre 5 y 8.5. Así que, monitorizando este valor, podremos ver la calidad de nuestros terrenos de plantación y, en caso de pH alto/bajo, poder arreglarlo con el conocimiento de la química necesario.

En tercer lugar se podría medir el nivel de CO_2 en el aire. En primera instancia, para establecer niveles óptimos de CO_2 en el lugar del cultivo. Cualquier incremento en la concentración ambiental de CO_2 aumenta la velocidad de la fotosíntesis y la cantidad total de azúcar producida por el cultivo, aunque esto no siempre tiene por qué ser bueno. Debemos considerar tres elementos: relación óptima de enriquecimiento para la cantidad de luz disponible, reacción del cultivo ante el aumento de dióxido de carbono suministrado y efectos del CO_2 adicional sobre el equilibrio y producción de la planta.

Por último, aunque la lista podría ser bastante extensa pero el alcance de este trabajo no es el cultivo real si no la parametrización del mismo (aunque si se demuestra el porqué de la monitorización de dichos parámetros y no otros), un detector de humos que, aunque no es un parámetro que pueda afectar directamente a la planta, sí que podría detectar un incendio. La mayoría de campos de cultivo se encuentran en las afueras (o, incluso, más lejos) de los centros urbanos, por lo que al no estar en el campo se hace casi imposible detectar un incendio a tiempo. Es por ello que, un detector de humos podría hacer que detectáramos un incendio en tiempo temprano y poder así evitar su expansión.

Capítulo 5. Elección de los sensores y acondicionamiento

Una vez explicadas, y principalmente entendidas, todas las magnitudes físicas a ser monitorizadas necesarias para el control de los parámetros que puedan afectar al cultivo en cuestión mencionado en el capítulo anterior, se procede a la elección de los sensores que van a ser utilizados para poder realizar las medidas oportunas y cumplir así el objetivo de tener el control sobre nuestro cultivo.

Para comenzar, se iniciará con la explicación más básica que se podría hacer en este apartado, la definición de un sensor. Un sensor es un objeto capaz de detectar las magnitudes físicas o químicas que se deseen (la temperatura, presión, desplazamiento, y todas las mencionadas en el capítulo 4 de esta memoria), dependiendo del tipo de sensor, y transformarlas en variables eléctricas cuantificables. Estas magnitudes físicas son también conocidas como variables de instrumentación.

Una vez introducido dicho concepto, se continuará con el desglose de todos los sensores que se van a utilizar para la realización de la estación sensora, dejando de lado el estudio previo y comenzando en este momento con el proceso de implementación de la misma. Cabe destacar que la plataforma utilizada para el desarrollo de hardware y software de este proyecto va a ser Arduino, compuesto por circuitos impresos que integran un microcontrolador y un entorno de desarrollo (IDE) en donde se procederá a programar la placa. Toda la lógica para obtener las mediciones será implementada en Arduino, pero la parte de toma de decisiones y recomendaciones no, si no que éstas serán intrínsecas a la aplicación Android.

5.1 DHT22: sensor de temperatura y humedad

Para poder realizar una estación sensora funcional y precisa a la hora de hacer mediciones, la elección acertada de los sensores pasa a ser el quid de la cuestión. Ni la mejor programación hará que un sistema funcione correctamente si el sensor de medida no es válido para los objetivos establecidos.

Como bien se ha explicado anteriormente, la temperatura y la humedad son dos magnitudes físicas clave para asegurar el futuro próspero de la planta y, a su vez, del fruto generado. Por ello, no se puede elegir cualquier sensor de temperatura y humedad al azar si no que, para tomar la decisión final, se ha buscado información previa, se ha hecho un estudio de objetivos a cumplir y se ha comprobado que las especificaciones del sensor cumplen con los objetivos establecidos.

En un primer momento, después de descartar varios, se quedaron tres en la lista de posibles sensores finales: DHT11 y DHT22 (digitales), y LM35 (analógico). Éste último fue descartado porque, a pesar de tener buena respuesta para la temperatura y su acondicionamiento es mínimo, el hecho de que los otros dos sensores nos den temperatura y humedad al mismo tiempo y tengan también una respuesta correcta ante dichas magnitudes, hacen que el tercero sea eliminado de la lista. Hay que recordad que, además de preciso, debe ser funcional y práctico, por lo que si un sensor puede hacer medidas de varios parámetros de forma precisa y simultánea, se descarta la posibilidad de tener varios sensores para el mismo fin (el espacio disponible es reducido). Con estos dos últimos sensores en lista, se procede a realizar una comparación más exhaustiva observada en la Tabla 4:

	DHT11		DHT22	
Rango de temperatura	0ºC ~ 50ºC		-40ºC ~ 80ºC	
Rango de humedad	20% ~ 90% HR		0% ~ 100% HR	
Precisión de la temperatura	± 2ºC		< ± 0.5ºC	
Precisión de la humedad	± 5HR		Media ± 2HR (máx. ± 5HR)	
Resolución de la temperatura	1ºC		0.1ºC	
Resolución de la humedad	1% HR		0.1% HR	
Tiempo de sensado	1 segundo		2 segundos	
Alimentación	3 Vdc ≤ Vcc ≤ 5 Vdc		3.3 Vdc ≤ Vcc ≤ 6 Vdc	
Consumo durante mediciones	Mínimo	Máximo	Mínimo	Máximo
	0.5 mA	2.5 mA	1 mA	1.5 mA
Consumo en stand-by	Mínimo	Máximo	Mínimo	Máximo
	100 uA	150 uA	40 uA	50uA
Precio	Sobre 1.80 €		Sobre 3€	

Tabla 4. Predicciones basadas en la presión atmosférica junto a la temperatura

Como bien se puede comprobar, a pesar de que el precio es algo más bajo, el DHT11 no cumple los objetivos de máxima precisión que estamos buscando. Por el contrario, el DHT22 es la evolución mejorada del anterior. Éste tiene un rango de temperatura y humedad bastante más amplio que el DHT11, lo que nos permitirá registrar temperaturas bajo cero y medidas de humedades más reales que abarcan desde el 0% hasta el 100%(el DHT11 no llega a esos rangos). Por otro lado, en el tema de precisión de medidas, se comprueba que el DHT22 es mucho más preciso y, además, mejora la parte de la resolución de la medida, pudiendo medir hasta un decimal de temperatura y humedad. Los consumos también son menores en este último, al igual que la alimentación máxima se ve mejorada debido a que el DHT11 se podía romper en caso de alimentación a 5V y aparición de un pico de tensión. La única desventaja que presenta el DHT22 respecto al otro, además del precio, es el tiempo de sensado, siendo de 2 segundos con respecto a 1 segundo que presenta el DHT11. Analizando los requerimientos y los objetivos, finalmente se ha decidido utilizar el DHT22 (Figura 13) debido a temas de precisión y rangos de medida; realmente, el tiempo de sensado no es un parámetro crítico para el alcance de este proyecto, debido a que 1 segundo de diferencia no será determinante para el desarrollo del cultivo.

DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND

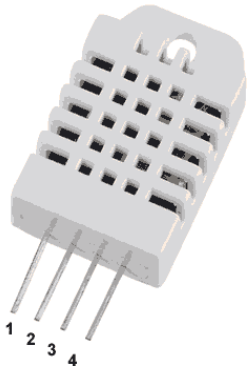


Figura 13. DHT22 (temperatura y humedad) y su relación pines-patillas

Como podemos comprobar, es un sensor bastante sencillo de entender. Solo tenemos que alimentarlo a 5V (o a 3.3V, pero recomendable 5V) por la patilla 1, conectarlo a tierra por la 4 y, por último, obtener los datos utilizando una salida digital de Arduino desde la patilla 2 del sensor. El significado de "NC" de la patilla 3 es No Connection, es decir, que esa patilla no se tiene que conectar a nada. En la versión más actualizada de este sensor (el que se va a utilizar) el sensor ya tiene solo 3 patillas. Hay que tener en cuenta que, a pesar de ser digital, este sensor necesitará una resistencia de pull-up situada entre la patilla de datos (2) y la alimentación (Vcc), asegurando los niveles lógicos necesarios para realizar el cambio (switch) de entrada a salida y viceversa (los pines son de tipo I/O); si no se pusiera esta resistencia, según especificaciones del catálogo ha de estar entre 4.7 kΩ y 10kΩ, el sensor podría no funcionar correctamente. En la Figura 14 podemos ver más gráficamente la conexión de este sensor con Arduino, siguiendo la explicación que se acaba de dar acerca de las patillas del sensor:

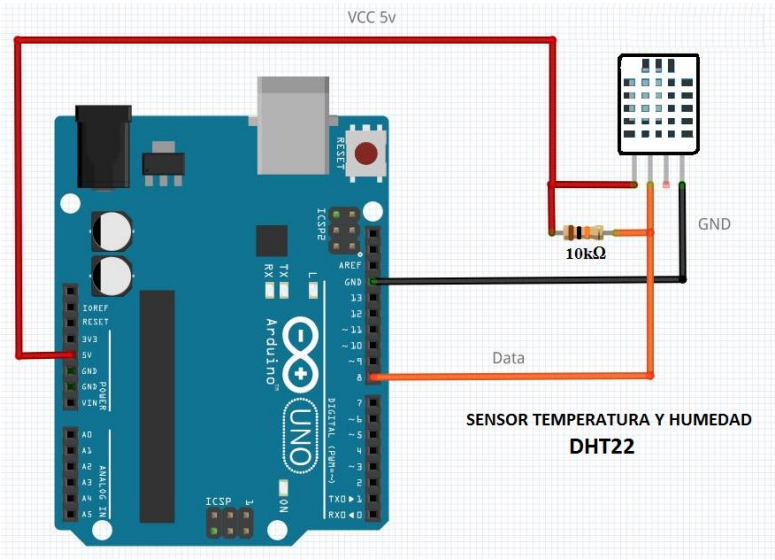


Figura 14. Acondicionamiento y conexión DHT22 con Arduino

Una vez realizada la conexión anterior, se procede a la programación de Arduino para poder obtener los datos de temperatura y humedad necesarios a partir del DHT22. Este sensor tiene una librería llamada "DHT.h" que deberemos añadir a nuestro programa, ya que esta librería posee las funciones ya definidas de lectura de humedad, temperatura (en grados centígrados y en grados Fahrenheit) así como los índices de calor. Una vez programada correctamente la función que nos dará los valores buscados, y añadiendo la librería anteriormente nombrada, se

procede a mostrar los resultados de temperatura y humedad relativa por el puerto serie, tal y como se puede ver en la Figura 16, junto a la programación de dicho sensor, en la Figura 15:

```
#include <DHT.h>
#include "DHT.h"
#define DHTPIN 8     // Pin digital conectado
#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  // Esperamos los 2 segundos necesarios para hacer cada medida
  delay(2000);

  float h = dht.readHumidity(); //Humedad
  float t = dht.readTemperature(); //Temperatura en Centigrados


  //COMPROBACIÓN EN ERRORES DE LA MEDIDA
  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;}

  Serial.print("Humedad: ");
  Serial.print(h);
  Serial.println(" %\t");
  Serial.print("Temperatura (C):");
  Serial.print(t);
  Serial.println(" *C\t ");
  Serial.println("");

}
```

Figura 15. Programación del sensor DHT22 en Arduino

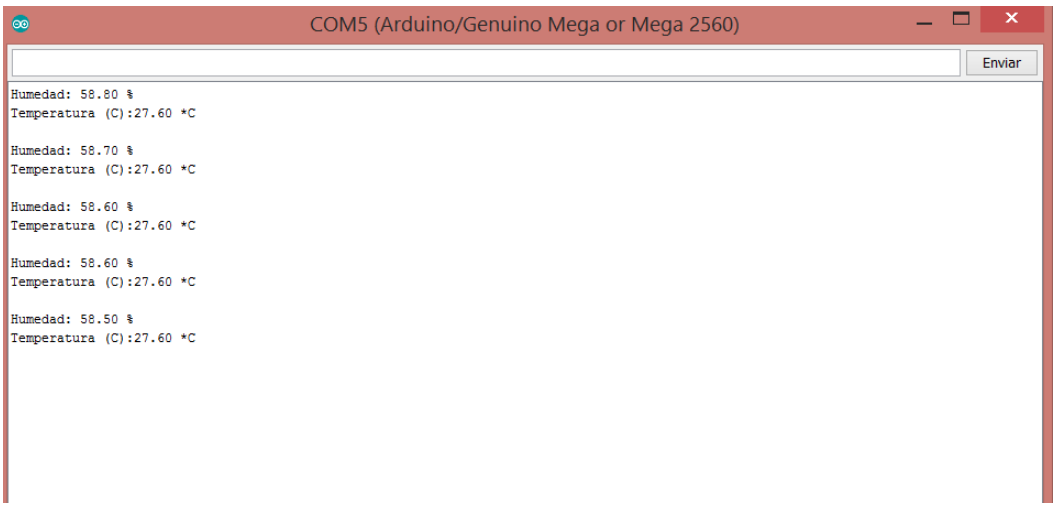


Figura 16. Salida del sensor DHT22 en Arduino

Los resultados se acercan mucho a la realidad ya que se dispone de un termohigrómetro comercial el cual marca 27.7ºC y 58.55% de humedad, llegando a la conclusión de que se han seleccionado buenos sensores inicialmente y, además, se ha programado correctamente.

Toda toma de decisiones respecto a qué se realizará en caso de altas/bajas temperaturas, al igual que con la humedad, se realizarán en la parte de la aplicación Android, explicada más adelante.

5.2 BPW21R: sensor de luz visible

El siguiente sensor a analizar es el encargado de captar la luz en el rango de frecuencia visible. Como se ha explicado en el apartado 4.3.1, para el caso de la tomatera el tiempo de exposición a la luz visible no es algo crítico, ya que no le afecta el fotoperiodo debido a que sus necesidades lumínicas oscilan entre 8 y 16 horas, pero sí la intensidad con la que llega. Por ello, se ha decidido comprobar si el nivel lumínico se encuentra por encima de un valor establecido y, si fuera así, notificarlo y evitar así que el nivel lumínico esté fuera del rango necesario. Además, será determinista al momento de saber si es buena hora para poder regar.

En este caso, por dar también un camino diferente a la electrónica digital, se ha decidido cuantificar esta magnitud física usando un sensor analógico para justificar este proyecto como un trabajo compuesto por electrónica analógica, además de la digital. Arduino está preparado para obtener valores tanto analógicos como digitales, ya que tiene entradas/salidas distinguidas para ambos casos.

Por razones de requerimientos acerca de dicho sensor, se ha decidido que sea poco directivo para abarcar más zona de medida, por lo que como mínimo se pide que el ángulo de media-sensibilidad, o dicho de otra manera, el ángulo en el cual la sensibilidad cae al 50%, sea ± 40º. Esto se hace para que mida una media de irradiancia en un ángulo sólido lo suficientemente grande como para que la zona de medida pueda captar los rayos solares oblicuos que se producen, por ejemplo, en invierno.

Finalmente se ha decidido medir esta magnitud física mediante el fotodiodo BPW21 (Figura 17). Este sensor cumple los requerimientos anteriormente mencionados. Su ángulo de media sensibilidad es de ± 50º, mejorando el mínimo previsto inicialmente. Su respuesta es muy lineal y tiene un encapsulado especialmente diseñado para aplicaciones lineales de alta precisión. Es capaz de captar radiación de luz visible desde 420 nm hasta 675 nm, rango aceptable para la aplicación a ejecutar ya que la sensibilidad relativa está por encima del 50% respecto del máximo de la misma, tal y como se comprueba en la Figura 18 (además de especificaciones del datasheet). El único problema es el precio que ronda los 9€ frente a otros sensores digitales más baratos e, incluso, fotorresistores que llegan a costar casi 10 céntimos la unidad. Pero, en ocasiones, un estudio de calidad (precisión, durabilidad, encapsulado) frente a precio es determinista, y en este caso así ha sido, siendo seleccionado un sensor cuya valía de trabajo supera el pesar de su precio.

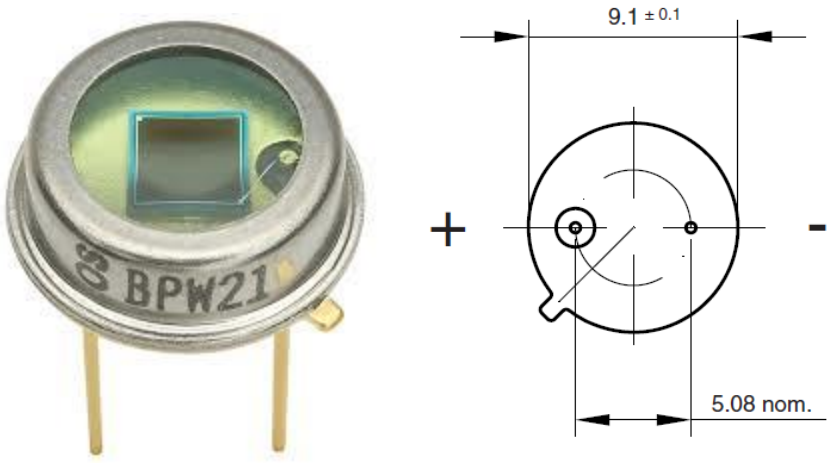


Figura 17. Sensor BPW21R (izquierda) y sus características (derecha)

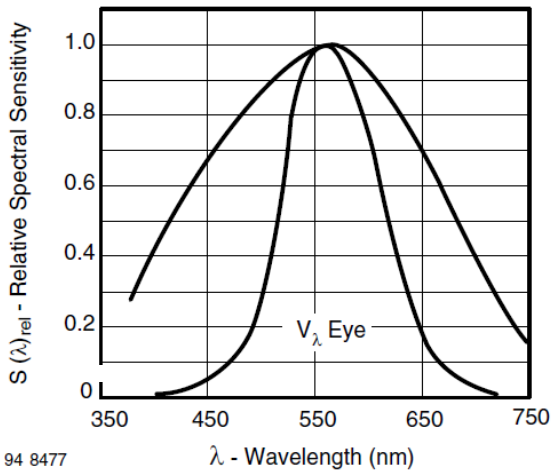


Figura 18. Sensibilidad espectral relativa vs longitud de onda (BPW21R)

Como se puede advertir, en la figura 17 se muestra la polaridad de las patillas del sensor, siendo la positiva la más cercana a la pestaña del encapsulado y la negativa la más alejada.

Para terminar de analizar sus características, se adjunta una figura más (Figura 19) la cual muestra gráficamente la sensibilidad relativa de radiación frente al ángulo con el que recibe el sensor dicha radiación:

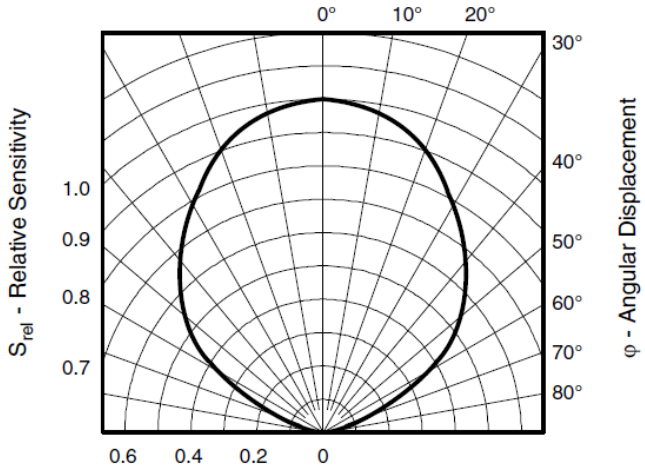


Figura 19. Sensibilidad relativa de radiación vs ángulo recepción BPW21R

Esta última figura nos permite comprobar lo ya citado acerca del ángulo de media sensibilidad mínimo, viendo más detalladamente en esta gráfica el comportamiento del sensor dependiendo del ángulo de desplazamiento o recepción, viendo que tiene una respuesta aceptable mientras el ángulo se mantenga entre ± 50º.

A continuación se comienza la parte de implementación de este sensor en Arduino. Al ser un sensor totalmente analógico, se debe justificar una parte de acondicionamiento de señal y un ajuste de los valores de las medidas realizadas con este sensor para obtener los resultados deseados a la salida.

Para la parte de acondicionamiento del fotodiodo, se ha seguido el siguiente esquema de conexión (Figura 20), conocimiento adquirido en la asignatura de Sensores, en el cual se necesitará un amplificador operacional, en adelante AO, siendo para este caso el TL081 un buen AO para la funcionalidad que queremos. Se ha utilizado realimentación negativa para poder justificar una mejor estabilidad del AO ya que, aunque se reduce un poco la ganancia, el circuito se vuelve más estable.

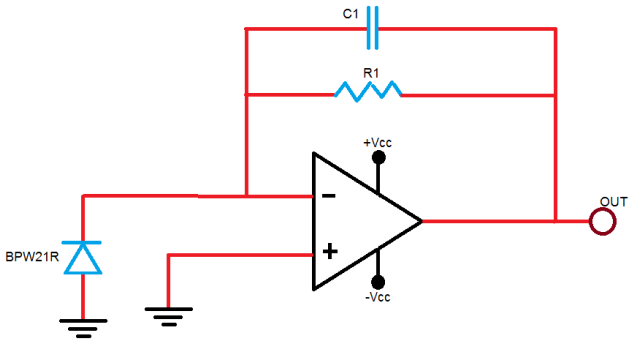


Figura 20. Esquema para el acondicionamiento del BPW21R con alimentación simétrica

La resistencia y el condensador que se ha puesto en la realimentación negativa (entre el terminal negativo y la salida) han sido colocados para asegurar una estabilidad en el sistema de medida y evitar los picos en las medidas de los valores, siendo estas más fiables. Para poder realizar esta función, se necesita que el valor de la resistencia sea bastante alta, en el orden de los cientos de kΩ, en nuestro caso se han cogido 200 kΩ; el condensador se ha elegido de 10 pF.

En esta fase del acondicionamiento, la alimentación necesaria para que funcione correctamente el AO tiene que ser simétrica, es decir, ±15V como máximo. Según el datasheet del TL081, la tensión de trabajo del AO va desde ±5V a ±15V, por lo que podremos alimentarlo con Arduino; pero el principal problema es que nosotros no tenemos alimentación simétrica, es decir, no tenemos los -5V. Para suplir dicho problema se ha realizado una transformación del esquema anterior para permitir alimentar dicho AO con +5V y 0V.

Una vez comprobado el funcionamiento del sensor a ±15V y, posteriormente, a ±5V, ambas utilizando la fuente de alimentación simétrica del laboratorio de la escuela, se procede a comprobar su funcionamiento para el caso de +5V (+) y 0V (-). Para ello, como bien se ha dicho, se va a seguir el esquema de la figura 20, pero haciendo una pequeña modificación en la entrada positiva del AO, quitando la conexión con tierra, y sustituyéndola por un divisor resistivo, tal y como se puede comprobar en la figura 21.

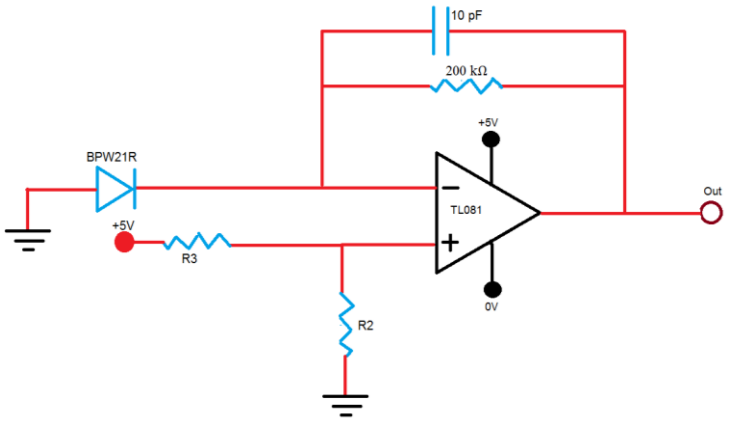


Figura 21. Esquema para el acondicionamiento del BPW21R con alimentación asimétrica (+5V/0V)

Como se puede comprobar en la figura anterior, se quedan dos parámetros sin definir: las resistencias R2 y R3. Para una primera verificación del nuevo sistema montado, se tomarán dos resistencias iguales cualesquiera, siendo las elegidas dos resistencias de valor 4.7 kΩ. Con ello, en el punto intermedio entre ambas resistencias, el punto conectado al terminal positivo del

TL081, tendremos una tensión de V/2, es decir, de 2.5V, tal y como se puede ver en la ecuación (5.2.1):

$$V_+ = Vcc * \frac{R2}{R1+R2} = 5 * \frac{4.7\text{k}\Omega}{4.7\text{k}\Omega+4.7\text{k}\Omega} = 2.5\ V \qquad \textbf{(5.2.1)}$$

Con las resistencias ya colocadas y con ayuda del osciloscopio, se procede a realizar un estudio de saturación para ver si funciona correctamente o, por el contrario, nos satura en valores que se querrían obtener. Al hacer una primera comprobación con luz artificial, vemos que satura muy pronto y no coge valores altos de luz, a los cuales nuestro cultivo estará expuesto. Se procede, pues, a ver la saturación tanto por arriba como por debajo de nuestro rango dinámico, desde 0 a 5V. Al comprobarlo mediante el osciloscopio, vemos que por arriba satura a un valor cercano a 4.2V y por debajo satura en 1.3V. Como lo que queremos es reducir el valor de entrada positiva en el AO, utilizaremos este valor mínimo de saturación, 1.4V como entrada del terminal positivo, teniendo pues un mayor rango dinámico de valores de luz y adaptándose a los esperados. En esta ocasión, tendremos que sustituir el valor de la R1, para obtener 1.4V en el punto intermedio de ambas resistencias, valor obtenido en la ecuación 5.2.2.

$$R_1 = R_2 * \frac{Vcc-V_+}{V_+} = 4.7\text{k}\Omega\ * \frac{5-1.4}{1.4} \approx 12\ \text{k}\Omega \qquad \textbf{(5.2.2)}$$

Finalmente, pues, se quedará el esquema de la Figura 22:

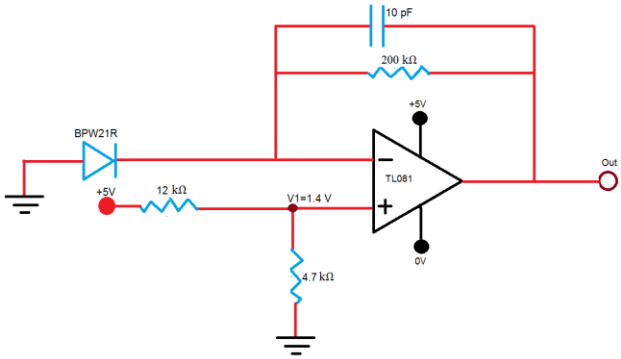


Figura 22. Esquema final para el acondicionamiento del BPW21R con alimentación asimétrica (+5V/0V)

Cuando teníamos las dos resistencias iguales de 4.7 kΩ, al hacer lecturas analógicas del sistema, saturaba con valores de luz medios. Al cambiar los valores por una de 12 kΩ y otra de 4.7 kΩ, las medidas ahora se ajustan a lo buscado, no saturando durante un día normal de luz. Esto se ha comprobado con una luz artificial de alta potencia alcanzando un valor analógico de unas 800 unidades analógicas de luz (valor que sale del sensor analógico), cuando en un día completo midiendo, dicho sensor alcanzó un valor máximo de 650 unidades de luz. Anteriormente, con ambas resistencias iguales, llegaba hasta los 430 y saturaba.

Por último, se procede a mostrar la programación utilizada para obtener esos valores de luz y, de nuevo, por el puerto serie, que irán cambiando conforme se acerque la luz artificial al sensor, todo esto disponible en la Figura 23 y la Figura 24. Hay que tener en cuenta que para conectarlo a Arduino, una vez realizado el esquema de la Figura 22, solo habrá que unir mediante un cable la patilla positiva del sensor BPW21 con una de las entradas analógicas del Arduino, en este caso la A3.

```
#define LVPin  A3


int miLV = 0;
void setup() {
  Serial.begin(9600);
}

void loop() {
 miLV = analogRead(LVPin)-340;  //Luz visible
  Serial.print("Luz visible: =  ");
  Serial.println(miLV);
  delay(1500);
}
```

Figura 23. Programación del sensor BPW21R en Arduino

No era necesario ponerle el delay de 1.5 segundos, pero se ha realizado para poder ir observando las medidas pausadamente y ver si corresponde de forma lógica con lo esperado. Por otro lado, se observa que a la lectura analógica del sensor se le resta 340; esto es debido a que se ha realizado un ajuste al término oscuro, es decir, se ha cogido y se ha aislado dicho sensor para que no reciba luz y se ha ajustado, de la mejor forma posible, dicho sensor para que frente a ausencia de luz dé un valor próximo a 0.

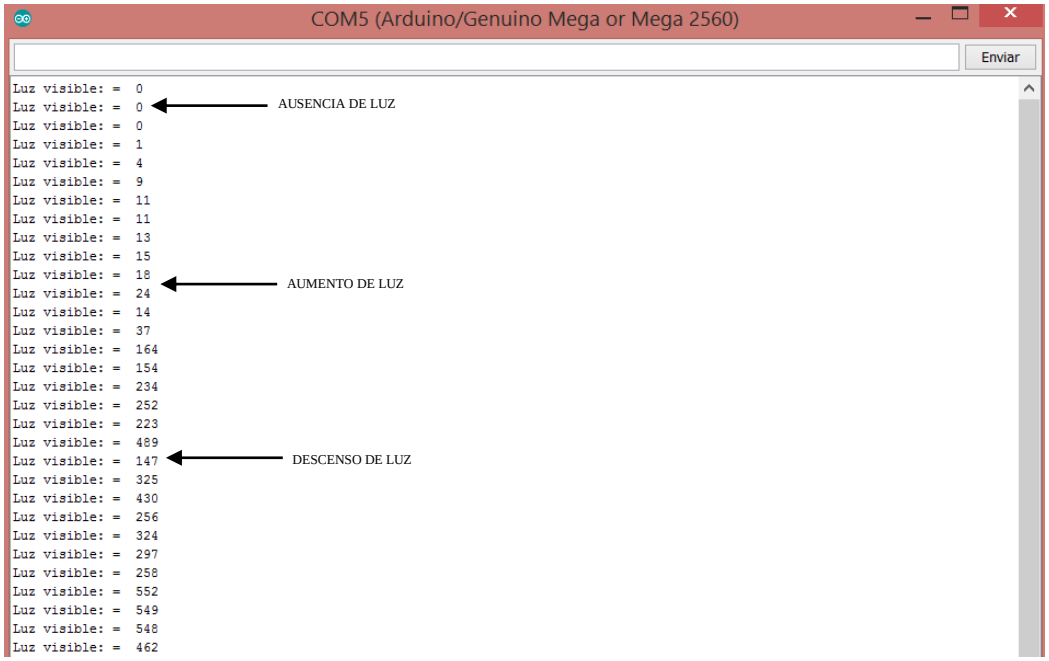


Figura 24. Salida del sensor BPW21R en Arduino

Como se puede comprobar, funciona correctamente, ya que en ausencia de luz (con un aislante lumínico) hemos conseguido que el nivel de luz visible sea 0 gracias al ajuste. Por otro lado, conforme vamos acercando una linterna (en esta ocasión, de poca potencia, de las pequeñas comerciales de led) vemos cómo va aumentando y disminuyendo progresivamente según se acerca o se aleja la luz.

De nuevo, la toma de decisiones se realizará desde la aplicación de Android.

5.3 BPV22F: sensor de radiación infrarroja

Al igual que en el apartado anterior se ha estudiado el sensor capaz de cuantificar el nivel de intensidad de luz visible, en éste se procede a realizar el mismo estudio pero, en esta ocasión, con respecto al sensor capaz de medir la radiación infrarroja. Como se explicó en su correspondiente apartado del capítulo anterior, un exceso de radiación infrarroja podría provocar que nuestra planta se seque y eche el producto a perder debido a que ésta está compuesta por una importante cantidad de agua. El agua absorbe muy bien las radiaciones infrarrojas lo que provoca un aumento de temperatura que elevará los grados de la propia planta hasta unos valores que, en caso de alta concentración de radiación infrarroja, ponen el peligro el desarrollo de la misma.

Tomando como referencia el caso de la luz visible, para estas medidas también se utilizará un sensor de tipo analógico, reforzando la parte analógica del proyecto y, de este modo, usando también las utilidades de Arduino como procesador digital/analógico.

En esta ocasión, se pide que el ángulo de captación de infrarrojos sea como mínimo ± 50º, algo más que el anterior debido a que el valor del nivel de infrarrojos en el ambiente es mucho más crítico que el valor de intensidad visible. Buscamos, otra vez, tener una superficie de medida lo suficientemente grande como para captar bastante información acerca del ambiente que rodea el cultivo.

El sensor seleccionado para cuantificar la radiación infrarroja ambiental ha sido el fotodiodo BPV22F que se observa en la Figura 25. Según especificaciones del datasheet, su ángulo de media-sensibilidad es de ± 60º, por encima del mínimo planteado inicialmente (± 50º). Por otro lado, es un sensor de alta velocidad con un tiempo de respuesta a la orden de los pocos nanosegundos y posee una gran sensibilidad a la hora de captar la radiación infrarroja con un rango de medida comprendido entre los 790 nm y los 1050 nm (donde la sensibilidad en estos extremos es superior al 50% del máximo de sensibilidad relativa), por tanto, fotodiodo admisible para la medida de este parámetro, pudiéndolo comprobar en la Figura 26, sacada directamente del datasheet, que relaciona la sensibilidad espectral relativa con la longitud de onda. La polaridad la podemos observar también en la misma figura, donde la patilla A corresponde a Ánodo (+) y la C a Cátodo (-).

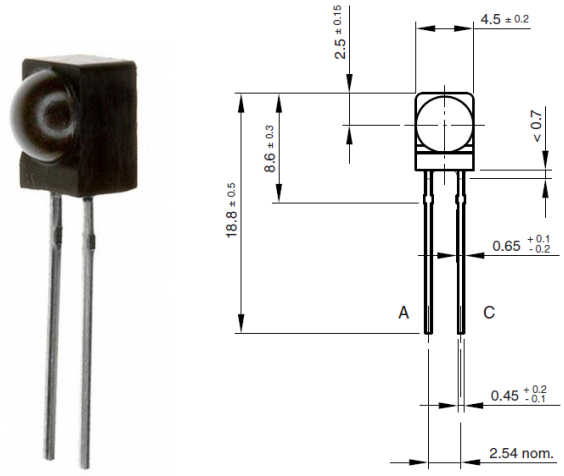


Figura 25. Sensor BPV22F (izquierda) y sus características (derecha)

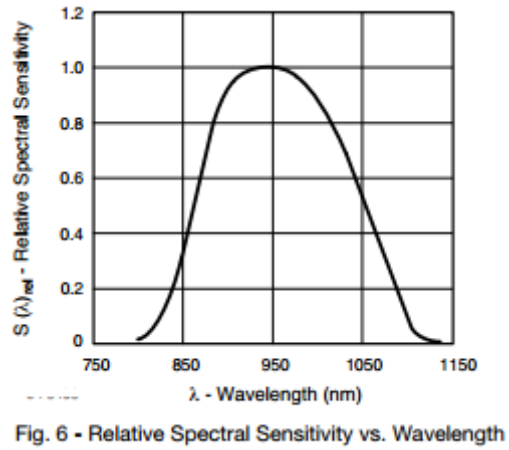


Figura 26. Sensibilidad espectral relativa vs longitud de onda (BPV22F)

Por último, en la Figura 27 podemos ver la relación entre la sensibilidad relativa de radiación en dependencia al ángulo de recepción o desplazamiento.

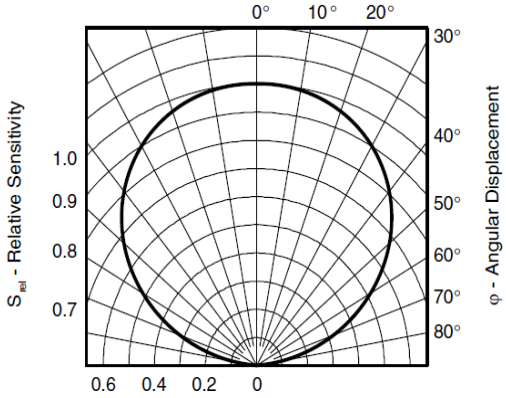


Figura 27. Sensibilidad relativa de radiación vs ángulo recepción BPV22F

Al igual que en el caso anterior con el sensor BPW21R, en este caso utilizaremos exactamente el mismo acondicionador, ya que se considera válido para ambos sensores de luz, independientemente de su franja frecuencial medida. Se puede ver el acondicionador usado en la siguiente figura (Figura 28):

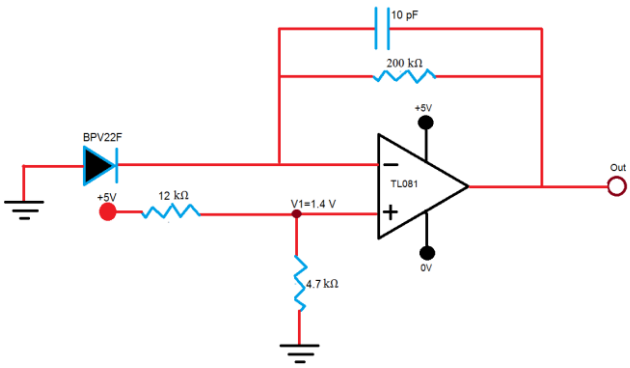


Figura 28. Esquema final para el acondicionamiento del BPV22F con alimentación asimétrica (+5V/0V)

Para acabar con este sensor, se procede a mostrar en la Figura 29 los valores de salida sacados por el puerto serie de Arduino. En esta ocasión se hace saber que la programación es la misma

que para el caso del sensor de luz visible, cambiando simplemente el ajuste a 0 y la conexión A3 con Arduino sustituida, por ejemplo, por la A4, pero manteniendo el resto del código.

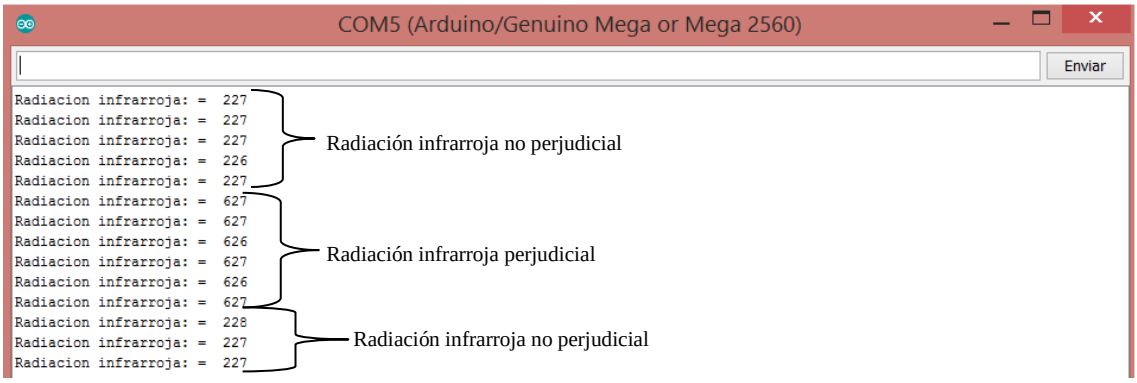


Figura 29. Resultado de medir la radiación infrarroja con el sensor BPV21F

Como se puede observar en la figura anterior, de normal se mide una radiación infrarroja por debajo de las 300 unidades de luz infrarroja (medida analógica); por el contrario, al aplicar una fuente de luz infrarroja, se observa una subida en este valor por encima de los 350 unidades de luz infrarroja. Por tanto, se establecerá este valor (350) como límite para determinar si la radiación infrarroja es perjudicial. Este valor no se ha elegido al azar, si no que mediante la medida de infrarrojos usando un termómetro de infrarrojos y la ayuda de un agricultor que lleva toda su vida junto a su padre cultivando (el cual nos ha prestado el termómetro y su ayuda incondicional), en consenso hemos decidido establecer ese valor midiendo los diferentes valores existentes a lo largo de su extensa granja e invernaderos.

5.4 TCS3200: sensor del color de la hoja

El siguiente sensor a estudiar es el sensor TCS3200, cuyo fin es obtener el color, en nuestro caso, de la hoja o del fruto. Su funcionamiento se basa en la conversión en frecuencia de la intensidad de luz medida por una matriz de fotodiodos. La frecuencia obtenida será mayor cuanta más grande sea la luminosidad que se detecte. Para poder compensar las diferencias de color en la superficie muestreada, la matriz de fotodiodos utiliza un promediado del valor medido.

La matriz de fotodiodos del TCS3200 está compuesta por 64 fotodiodos en total repartidos en 16 fotodiodos para el filtro rojo, 16 para el azul, otros 16 para el verde y, los 16 restantes, están sin filtrar. Hay que tener en cuenta que todos no funcionan de manera simultánea, si no que se produce una activación por filtros, es decir, por grupos de color antes de realizar la medida de la intensidad de luz incidente sobre ellos. Se dispone de dos patillas en el sensor, S2 y S3, que sirven para configurar en qué momentos se ha de activar cada filtro.

Una vez conseguida la lectura del nivel de iluminación en forma de corriente, ésta se convierte a frecuencia en forma de onda cuadrada con un ciclo de trabajo del 50%. Esto es debido a que es mucho más estable enviar al microcontrolador una frecuencia en vez de una intensidad, soportando así mejor las interferencias producidas por las pistas del circuito. Es posible escalar dicha frecuencia en tres niveles: al 100%, al 20% y al 2%. En este proyecto, se va a coger la frecuencia escalada al 2% para poder obtener los valores RGB del color. Tanto para seleccionar el filtro, como para seleccionar la escala aplicada, se seguirán las tablas vistas en la Figura 30, sacada directamente del datasheet del módulo:

S0	S1	OUTPUT FREQUENCY SCALING (f_o)
L	L	Power down
L	H	2%
H	L	20%
H	H	100%

S2	S3	PHOTODIODE TYPE
L	L	Red
L	H	Blue
H	L	Clear (no filter)
H	H	Green

Figura 30. Selección del filtro y el escalado a aplicar

Una vez introducida la forma de funcionamiento y trabajo de este sensor, se procede a mostrar al mismo, así como su esquemático para observar sus patillas y cómo deben conectarse, en la Figura 31:

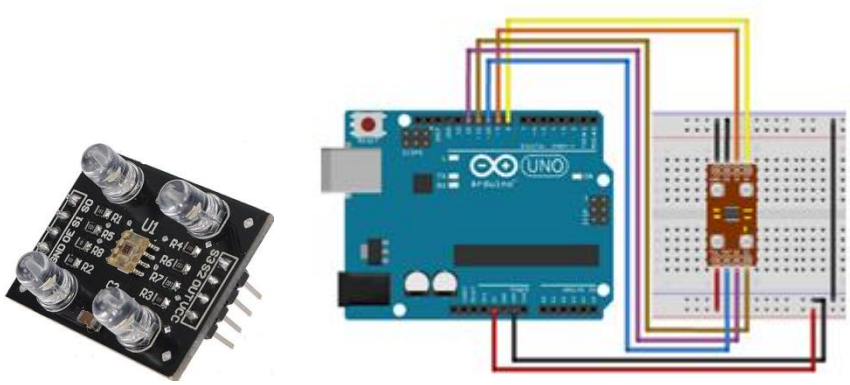


Figura 31. Sensor TCS3200 para medida del color (izquierda) y su conexión con Arduino (derecha)

Como se puede observar, los pines del S0 al S3 van conectados a las entradas/salidas digitales de Arduino. Por otro lado, el pin OE (Output Enable) debe ir conectado a GND debido a que el modulo va a enviar información continuamente al Arduino. Por último, OUT también va a ir a otra salida/entrada digital del Arduino. Teniendo todo conectado, y habiendo realizado un estudio teórico previo, se procede a mostrar la programación utilizada y a su posterior explicación, en varias figuras debido a la extensión del programa. Primero, se mostrará una primera versión del mismo, visto en las Figuras 32, 33, 34 y 35:

```
#include <TimerOne.h>

#define S0     3
#define S1     4
#define S2     5
#define S3     6
#define OUT    2
//INICIALIZACIÓN DE VARIABLES
int   s_count = 0;     // CONTADOR DE FRECUENCIA
int   s_array[3];      // VALOR RGB
int   s_flag = 0;      // filtros de RGB
float s_SF[3];         // factor de escala para RGB
int j=0;

// Inicialización TSC3200
void TSC_Init()
{
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
  pinMode(OUT, INPUT);
//AJUSTE DE FRECUENCIA; FRECUENCIA DE SALIDA ESCALADA AL 2%
  digitalWrite(S0, LOW);
  digitalWrite(S1, HIGH);
}

void TSC_Count()
{
  s_count ++ ;
}
```

Figura 32. Programación del sensor TCS3200 (primera parte)

En esta primera parte se observa la definición de las variables necesarias, así como las correspondencias de conexión entre los pines (S0-S3, OUT) y Arduino, además de la necesidad de incluir la librería “TimerOne.h” para poder utilizar los contadores en Arduino. Por otro lado, se observan 2 funciones: “TSC_Init()” encargada de la inicialización del TSC3200,

estableciendo el modo de los pines (OUTPUT o INPUT) y ajustando la escala de la frecuencia de salida; la otra función es "TSC_Count()" que como se puede intuir, se trata de un contador simple.

```
void TSC_Callback()
{
  switch(s_flag)
  {
    case 0:
         Serial.println("->WB Start");
         TSC_WB(LOW, LOW);              // Filtrado del rojo
         break;
    case 1:
         Serial.print("->Frequency R=");
         Serial.println(s_count);
         s_array[0] = s_count;
         TSC_WB(HIGH, HIGH);            //Filtrado del verde
         break;
    case 2:
         Serial.print("->Frequency G=");
         Serial.println(s_count);
         s_array[1] = s_count;
         TSC_WB(LOW, HIGH);             //Filtrado del azul
         break;

    case 3:
         Serial.print("->Frequency B=");
         Serial.println(s_count);
         Serial.println("->WB End");
         s_array[2] = s_count;
         TSC_WB(HIGH, LOW);             // Clear (sin flitrar)
         break;
   default:
         s_count = 0;
         break;
  }
}

void TSC_WB(int Level0, int Level1)      // Balance de Blanco
{
  s_count = 0;
  s_flag ++;
  TSC_FilterColor(Level0, Level1);
  Timer1.setPeriod(1000000);             // seleccionamos 1 segundo como periodo
}
```

Figura 33. Programación del sensor TCS3200 (segunda parte)

En esta segunda parte, de nuevo, se observan otras dos funciones: TSC_Callback() encarga de administrar y presentar los distintos casos de filtrado (verde, rojo, azul y sin filtrar) dentro de un switch dependiente del valor de s_flag, que inicialmente vale 0. Como podemos ver, inicialmente filtrará el rojo, luego el verde, posteriormente el azul y finalmente el "clear o sin filtrar" permitiendo obtener la frecuencia del color según la escala RGB. El aumento del valor de s_flag se da en la segunda función, TSC_WB que se encarga de poner el contador a 0 y aumentar el valor del flag, además de llamar a otra función, TSC_FilterColor que la veremos en la siguiente figura, y se encarga de la selección del filtro (rojo, verde, azul o clear).

```
// Selección del filtro de "" color
void TSC_FilterColor(int Level01, int Level02)
{
  if(Level01 != 0)
    Level01 = HIGH;

  if(Level02 != 0)
    Level02 = HIGH;

  digitalWrite(S2, Level01);
  digitalWrite(S3, Level02);
}

void setup()
{
  TSC_Init();
  Serial.begin(9600);
  Timer1.initialize();             // defaulte is 1s
  Timer1.attachInterrupt(TSC_Callback);
  attachInterrupt(0, TSC_Count, RISING);

  delay(4000);

  for(int i=0; i<3; i++)
    Serial.println(s_array[i]);

  s_SF[0] = 255.0/ s_array[0];     //R Scale factor
  s_SF[1] = 255.0/ s_array[1] ;    //G Scale factor
  s_SF[2] = 255.0/ s_array[2] ;    //B Scale factor

  Serial.println("Calibrando");
  Serial.println(s_SF[0]);
  Serial.println(s_SF[1]);
  Serial.println(s_SF[2]);
  Serial.println("CALIBRADO");
  Serial.println("");
}
```

Figura 34. Programación del sensor TCS3200 (tercera parte)

En la tercera parte de este programa, otra vez, se observan dos funciones más: TSC_FilterColor(int level01, int level02) la cual, como bien hemos dicho antes, se encarga de establecer, dependiendo de los valores de entrada, el filtro del color. Por otro lado, en la función setup (función la cual se ejecuta al principio y una sola vez, función de inicialización o establecimiento) se produce la inicialización del sistema con la llamada de TSC_Init(), así como una primera llamada a la función TSC_Callback la cual se usa para calibrar el sistema. En este momento inicial, el sensor se debe calibrar con el color blanco (por ejemplo, poniendo el sensor frente a un folio blanco). Todo el código siguiente se usa para poder calibrar el sistema con ese blanco, sacando un factor de escala que permitirá obtener los colores correctamente.

```
void loop()
{
   s_flag = 0;
   for(int i=0; i<3; i++)
   {
   if (j==2){
    Serial.println("Cantidad de azul = ");
  // Serial.println(int(s_array[i] * s_SF[i]));
   int azul = (s_array[i] * s_SF[i]);
   Serial.println(azul);
   j=0;
   }
   else if (j==1){
     Serial.println("Cantidad de verde = ");
  // Serial.println(int(s_array[i] * s_SF[i]));
   int verde = (s_array[i] * s_SF[i]);
   Serial.println(verde);
   j++;
   }
   else if (j==0){
     Serial.println("Cantidad de rojo = ");
   //Serial.println(int(s_array[i] * s_SF[i]));
   int rojo = (s_array[i] * s_SF[i]);
   Serial.println(rojo);
   j++;
   }
   }
  Serial.println("");
    delay(4000);

}
```

Figura 35. Programación del sensor TCS3200 (cuarta parte)

Por último, en el loop se irá realizando todo lo anterior citado, es decir, se irá comprobando el color mediante el TSC_CallBack y se sacará, mediante las medidas oportunas, el color que se encuentre en la superficie de medida del TSC3200, cuyos resultados se pueden ver en la Figura 36:

```
->WB Start
->Frequency R=1140
->Frequency G=951
->Frequency B=1073
->WB End
1140
951
1073
Calibrando
0.22
0.27
0.24
CALIBRADO

Cantidad de rojo =
255
Cantidad de verde =
255
Cantidad de azul =
255
```

```
->WB Start
->Frequency R=864
->Frequency G=618
->Frequency B=405
->WB End
Cantidad de rojo =
193
Cantidad de verde =
165
Cantidad de azul =
96

->WB Start
->Frequency R=545
->Frequency G=632
->Frequency B=404
->WB End
Cantidad de rojo =
121
Cantidad de verde =
169
Cantidad de azul =
96
```

Figura 36. Calibrado en blanco (izquierda) y medida (derecha) del sensor TSC3200.

Como se puede comprobar, lo primero que se produce es la calibración del sensor con respecto al blanco (R=255, G=255 y B=255) y, luego, comienza a medir (en el ejemplo de la Figura 36 derecha, primero marrón y luego verde).

Por último nos queda el tema de seleccionar el tipo de hoja según el color. Como solo disponemos de un sensor TSC3200, solo estudiaremos el color de la hoja, pero se hace saber que, con otro sensor más, se podría estudiar simultáneamente el color del fruto también. En la Figura 37 se puede ver la clasificación de las hojas según su código RGB. El estudio para su clasificación se ha realizado con 10 hojas diferentes, 50 medidas de cada hoja y la ayuda de Excel. Se considera suficiente para el alcance de este trabajo, aunque se recomienda recalibrar según tipo de cultivo e incluso realizar una posible revisión en el caso de que se mantenga el cultivo de la tomatera.

```
if (rojo>10 && rojo<20 && 100<verde && verde<150 && azul<50){ //Hoja sana tipo I
  tipoHoja=1;
  Serial.print("Tipo hoja: ");
  Serial.println(tipoHoja);
}
else if (rojo<10 && 50<verde && verde<120 && azul<30){ //Hoja sana tipo II
  tipoHoja=2;
    Serial.print("Tipo hoja: ");
  Serial.println(tipoHoja);
}
else if (40<rojo && rojo <50 && 180<verde && verde<230  && azul<30){ //Hoja sana con falta de nutrientes, revisión
  tipoHoja=3;
  Serial.print("Tipo hoja: ");
  Serial.println(tipoHoja);
}
else if (40<rojo && rojo<50 && 120<verde && verde<150 && azul<20){ //Hoja con falta de agua
  tipoHoja=4;
  Serial.print("Tipo hoja: ");
  Serial.println(tipoHoja);
}
else if (rojo>190 && verde >190 && azul<50){ //HOJA SECA, ALARMA!
  tipoHoja=5;
  Serial.print("Tipo hoja: ");
  Serial.println(tipoHoja);
}
else if (rojo>190 && verde <150 && azul<50){ //HOJA MUERTA!
  tipoHoja=6;
  Serial.print("Tipo hoja: ");
  Serial.println(tipoHoja);
} else{
  tipoHoja=7; //hoja no registrada, para revisión
  Serial.print("Tipo hoja: ");
  Serial.println(tipoHoja);
}
```

Figura 37. Programación para la clasificación de las hojas según su color RGB

De nuevo, se considerará esta clasificación suficiente para a lo que este proyecto se refiere, aconsejando el reajuste para futuras medias. El tipo de hoja (dato numérico) será lo que se pasará como dato para subir a la base de datos debido a que el color exacto de la hoja no es de importancia, pero si su estado determinado a partir de este color. Por eso, se han tomado dos estados de hoja sana, otro de falta de nutrientes, falta de agua, hoja seca y hoja muerta. El resto de casos, además del caso de falta de nutrientes, será llevado a revisión por el agricultor personalmente. Los otros se dejan a elección del mismo. El resultado en esta ocasión será el visualizado en la Figura 38.

```
Cantidad de rojo =
15
Cantidad de verde =
120
Cantidad de azul =
10
Tipo hoja: 1

->WB Start
->Frequency R=368
->Frequency G=394
->Frequency B=294
->WB End
Cantidad de rojo =
205
Cantidad de verde =
193
Cantidad de azul =
38
Tipo hoja: 5
```

Figura 38. Clasificación de las hojas por tipos

Como se puede comprobar, después de la calibración, se le puso una hoja sana de tomatera recién cortada y nos dio hoja de tipo 1 (hoja sana). Rápidamente, cambiamos el tipo de hoja a una que se había dejado secar un tiempo y, nuevamente, acertó, concluyendo así este apartado.

5.5 YL-69 e YL-38: sensor de la humedad del terreno

Otro parámetro importante a medir, como se comentó en el capítulo de magnitudes físicas, es la humedad existente en el terreno. Para ello, el sensor que hemos elegido para poder cuantificar esta magnitud es el conjunto formado por los módulos YL-69 e YL-38.

La importancia de la medida de la humedad del terreno proviene de la necesidad de saber en todo momento si nuestro cultivo necesita ser regado o no, además de no regar en caso de predecir lluvias abundantes (usando el sensor de presión atmosférica y el de temperatura). Es, por tanto, un punto crítico a la hora de determinar si ese cultivo necesita agua y, además, fundamental si en un futuro se quisiera domotizar el regado del campo.

Como bien se ha introducido, el sensor de humedad del terreno es el compuesto por el módulo YL-69 y el módulo YL-38, que pueden ser vistos en la Figura 39.

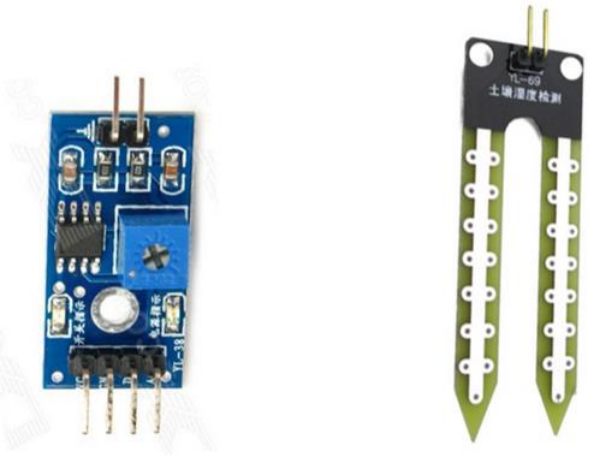


Figura 39. YL-38 (izquierda) e YL-69 (derecha), en conjunto, sensor humedad del terreno

Por describirlos brevemente, el YL-69 es un módulo/sensor que posee la capacidad de medir la humedad del suelo. Esto se logra gracias a la aportación de una pequeña tensión entre los terminales de dicho módulos, haciendo circular una corriente cuyo valor dependerá de la resistencia que genera el suelo, dependiendo en su mayor parte de la humedad (además de minerales, que para el alcance de este proyecto lo consideraremos como nulo frente a la humedad del terreno). Se sobreentiende que, al aumentar la humedad, la corriente se hace más grande; por el contrario, al bajar la humedad, la corriente disminuye.

Por otro lado, el YL-38 no es más que un comparador LM393, que consiste en dos comparadores de tensión y, por tanto, será usado para determinar el valor de humedad del terreno en cuestión.

El sistema al completo, que forman el sensor funcional, consiste en una sonda YL-69 con dos terminales adecuadamente separados y un módulo YL-38 con el comparador mencionado. Este último dispone de dos pines de conexión hacia el otro módulo, otros 2 pines para la alimentación Vcc (5V) y tierra (GND) y 2 pines más para salida de datos (D0 Digital y A0 Analógica). En nuestro caso vamos a usar todas las patillas, exceptuando la patilla digital, ya que vamos a sacar el valor de forma analógica. Por otro lado, no se precisa de acondicionador

para este sensor. La conexión entre módulos, así como con el Arduino, se puede observar en la Figura 40:

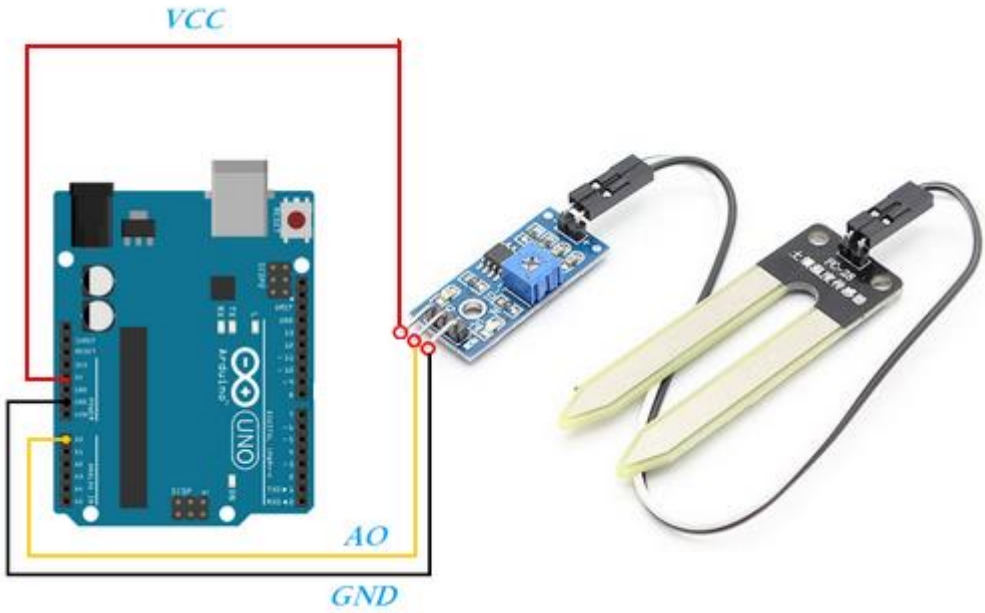


Figura 40. Conexión de los módulos YL-69 e YL-38 con Arduino

Una vez conocido el conexionado a ejecutar, se procede a realizar la programación necesaria para obtener a la salida los valores de humedad del terreno y, además, poder clasificarlos en 4 estados diferentes: fuera de la tierra o no conectado, seco, húmedo y muy húmedo o en el agua. Esta clasificación se ha hecho, al igual que en el caso del sensor del color, porque este sensor no te dice exactamente la humedad del terreno, si no que devuelve valores analógicos que se han de interpretar mediante un estudio previo realizado y su posterior ajuste para un buen entendimiento del valor. Es por ello que se ha decidido subir a la base de datos un número del 1 al 4, dependiendo de la clasificación. Así, no se leerá un valor de humedad de terreno, si no un número que determinará si el suelo estará seco, húmedo o cualquier otra opción. Se procede, pues, a mostrar en la Figura 41 la programación realizada para este sensor. También, en la Figura 42, se podrá ver el resultado/salida por el puerto serie de dicha programación.

```
void setup() {
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

void loop() {
 int humTer = analogRead(A0);
  int humTersal=0;
   if(humTer >= 1000) {
    humTersal=1; //Desconectado o no en tierra
    Serial.println("Sensor desconectado o fuera de tierra");
    Serial.println(" ");
  }
  if(humTer < 1000 && humTer >= 600) {  //Seco
   humTersal=2;
  Serial.println("Terreno seco. Se recomienda regar si prevision meteorologica no dice lo contrario");
  Serial.println(" ");
  }
  if(humTer < 600 && humTer >= 370) { //humedo
   humTersal=3;
  Serial.println("Terreno humero. NO REGAR.");
  Serial.println(" ");
  }
  if(humTer < 370) { //en el agua/demasiado humedo
   humTersal=4;
   Serial.println("Terreno acuoso o en agua. ALERTA, posibilidad de muerte de planta si no se está regando en este instante");
  Serial.println(" ");
  }
  delay(2000);
}
```

Figura 41. Programación del sensor de humedad de terreno (YL-69+YL-38)

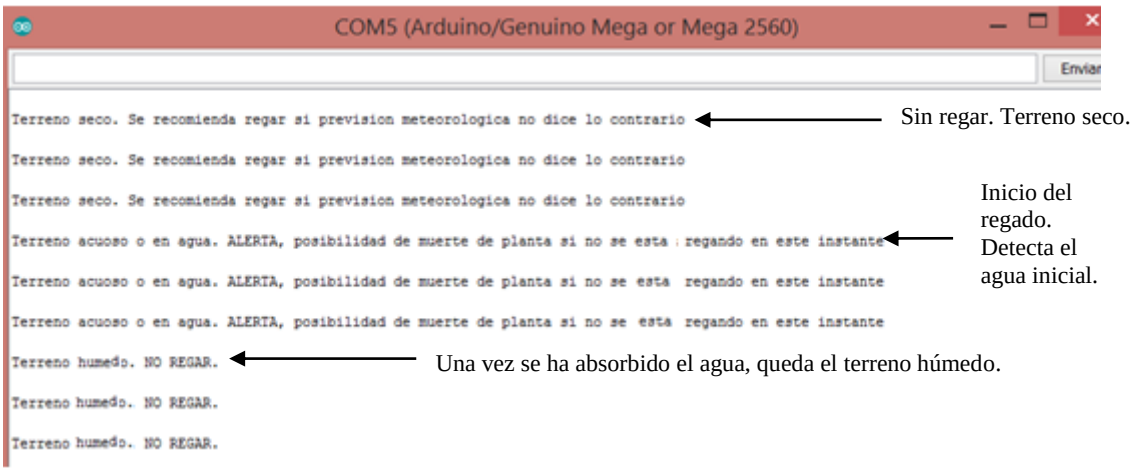


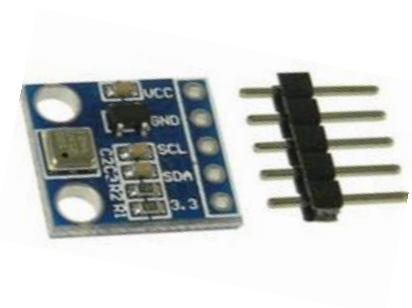
Figura 42. Resultados del sensor de humedad de terreno (YL-69+YL-38)

Como se puede ver en la figura anterior, en un primer momento metimos el sensor en la tierra del cultivo seca, es por ello que marca "terreno seco" y que "se recomienda regar". Una vez se riega, debemos tener conocimiento acerca de si se produce el regado. Si aparece el cartel de "terreno acuoso" y no estamos regando (o no está lloviendo), se ha producido una inundación en el cultivo que puede destrozarlo. En caso contrario, si se riega o llueve, entonces significa que está detectando el agua que aún no se ha absorbido. Por último, una vez absorbida el agua, marca "terreno húmedo" como era de esperar.

5.6 BMP180: sensor de presión atmosférica

Por último, se finalizará con el sensor encargado de medir el parámetro relativo a la presión atmosférica. Para poder realizar esta medida de forma precisa y esperando resultados fiables, se ha seleccionado el sensor BMP180, definido como un sensor de alta precisión, el cual cuenta en todas las tiendas de ventas con una alta posición en correspondencia a los comentarios que los compradores dejan, además de lo que se puede leer de las especificaciones, siendo pues motivo de elección del mismo.

Por definición, el módulo BMP180, que se puede ver en la Figura 43 junto a su correspondencia de pines con Arduino, es un sensor que nos permite medir la presión atmosférica (también conocida como barométrica) y, como uso adicional que no se utilizará para este proyecto pero se deja en constancia por si en un futuro se quisiera usar, diferencia de alturas entre dos puntos.



BMP180	ARDUINO
VCC	5V
GND	GND
SCL	SCL(21)
SDA	SCL(20)
3.3 (si no VCC)	3.3V

Figura 43. Sensor BMP180 con su correspondencia de pines con Arduino

Además de la correspondencia de pines, se adjunta en la Figura 44 la conexión exacta para que no quepa duda acerca de la misma. Como se ha podido comprobar, se ha decidido cambiar de Arduino, pasando del Uno al Mega, debido a que se usan demasiados sensores y la memoria se llena con facilidad en el caso del primero. Ninguno de los sensores anteriores sufre ninguna modificación de conexión. Como se puede ver en la figura, todo se ha conectado según lo especificado en la tabla de la Figura 43. Para poder usar este sensor, existe una librería para Arduino llamada "BMP180.h" que debemos añadir a nuestro programa.

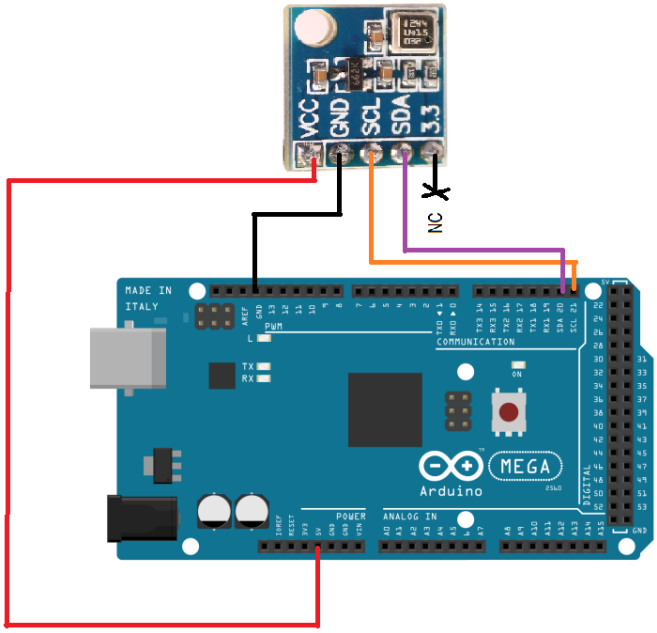


Figura 44. Conexión sensor BMP180 con Arduino

Como bien se puede comprobar, necesita los pines SCL y SDA del Arduino. Por hablar de forma resumida de estos pines, el pin SCL es la línea por donde va la señal para sincronizar las transferencias de datos (reloj) y el pin SDA es la línea por la cual se transfieren los datos, siguiendo el formato (| start | A7 A6 A5 A4 A3 A2 A1 R/W | ACK |… DATA…| ACK | stop |idle|). Por lo tanto se sobreentiende que el BMP180 irá enviando según le marque el reloj de sincronización de transferencia de datos.

Por eso, y en resumen, se puede decir que el sensor BMP180 es capaz de medir la presión barométrica relativa (en pascales, aunque se ha decidido pasar a milibares para compararlo con las medidas normales del Aemet, siendo su paso tan sencillo como dividir entre 100 el valor de los pascales).

Por último, en la Figura 45 y la Figura 46 se puede ver, respectivamente, la programación utilizada para este sensor, así como la salida del sensor en el puerto serie del Arduino. Como se ha dicho en párrafos anteriores, se necesita incorporar la librería "BMP180.h", realizar la llamada a la medida que marca su datasheet para el caso de la presión atmosférica y, en nuestro caso, transformar la medida de pascales a milibares. Se dejará 2 segundos entre medida y medida para ajustarse al tiempo de reloj que se ha mencionado anteriormente.

```
#include <Wire.h>
#include <Adafruit_BMP085.h>

Adafruit_BMP085 bmp;
 float pa;
 float mbar;
void setup() {
  Serial.begin(9600);
  if (!bmp.begin()) {
  Serial.println("Sensor no conectado");
  while (1) {}
  }
}

void loop() {
   // Serial.print("Presion atmosferica(Pa) = ");
    pa=bmp.readPressure();
   // Serial.print(pa);
   // Serial.println(" Pa");
    mbar=pa/100;
    Serial.print("Presion atmosferica(mbar) = ");
    Serial.print(mbar);
    Serial.println(" mbar");

    Serial.println();
    delay(2000);
}
```

Figura 45. Programación del sensor BMP180

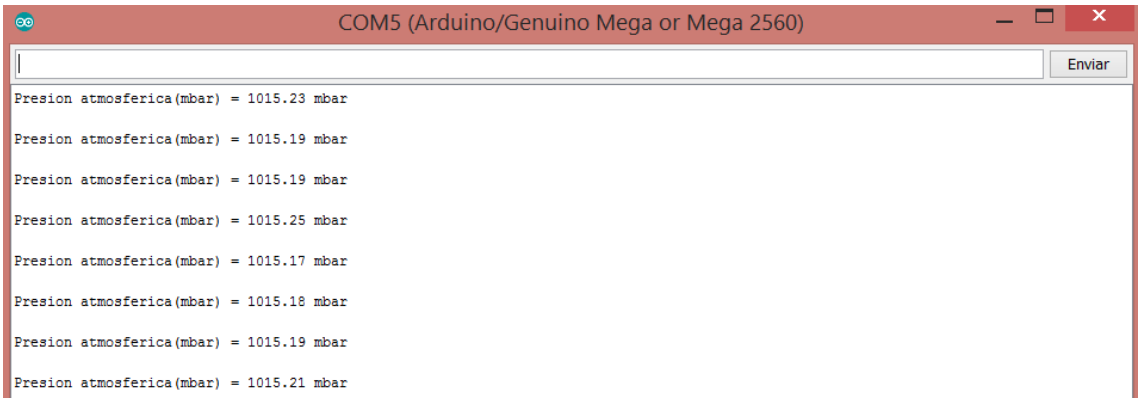


Figura 46. Salida del sensor BMP180

Comprobando el mapa de presión atmosférica del día de realización de este proyecto (Figura 47) se observa que estamos dentro de la zona de los 1015 mbar de presión, por lo que podemos decir que el sensor, así como su medida realizada, funciona correctamente.

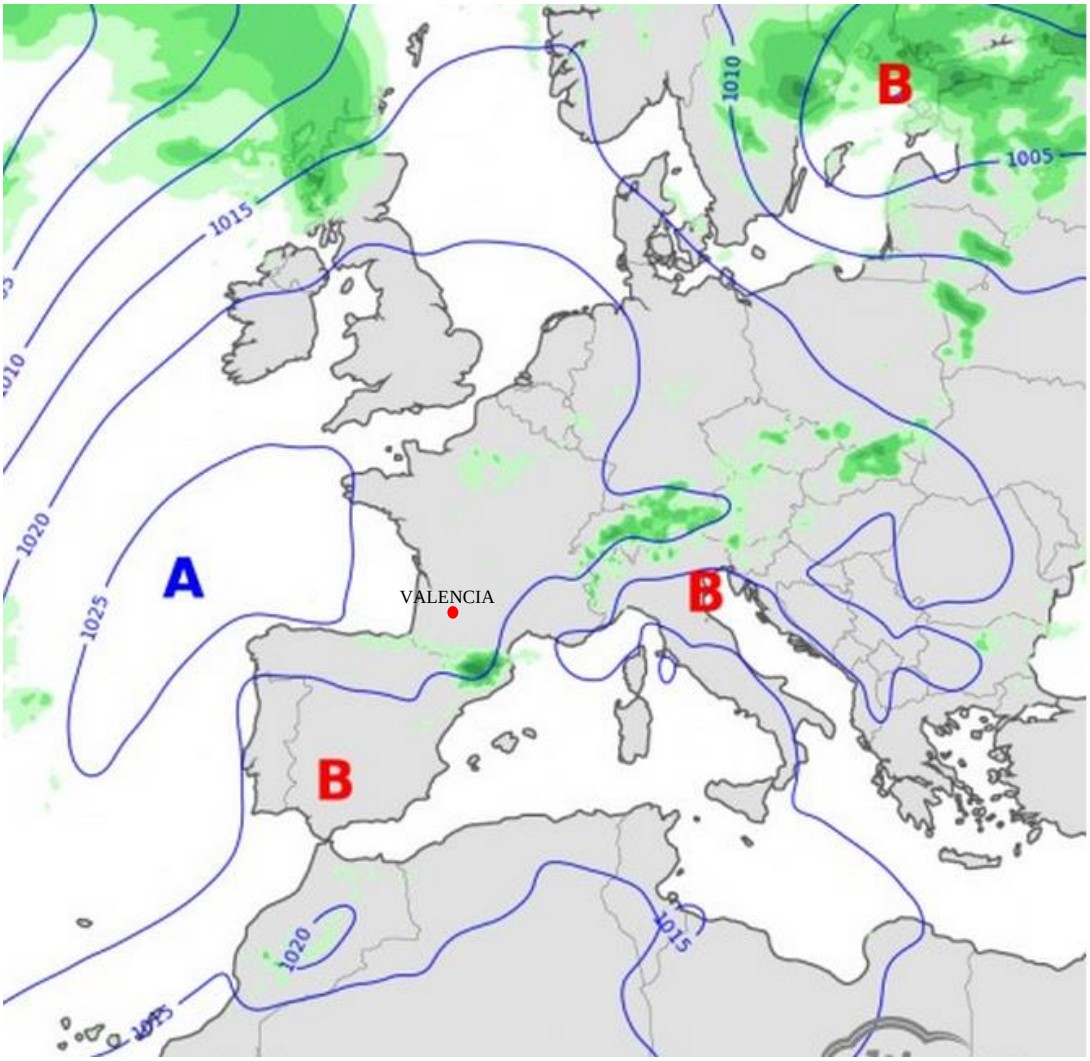


Figura 47. Presión atmosférica existente el día de la medida

Capítulo 6. Descripción de la estación de sensores

En los dos capítulos anteriores se ha hecho, en primer lugar, una introducción de las magnitudes a medir por su criticidad en los cultivos y, posteriormente, se ha realizado la selección de los sensores más óptimos para realizar dichas medidas, así como las explicaciones pertinentes acerca de la conexión y la programación de los mismos en Arduino.

Ahora, se procede a realizar una vista general de la unidad central de este trabajo, la estación de sensores, entendida como la conexión del Arduino con todos los sensores simultáneamente y, obviamente, la adaptación de la programación cogiendo todos los programas individuales de los sensores en cuestión y, empleando la lógica y buena praxis de programación, uniéndolos en un único código en lenguaje Arduino.

En la Figura 48 se puede observar el módulo (como conjunto) de la estación de sensores formada por la placa de conexión, donde estarán conectados los sensores y sus respectivos acondicionadores, junto al módulo de Arduino Mega, que será el encargado de procesar todas las medidas. En un primer vistazo, se puede llegar a la conclusión de que se podría mejorar el orden e, incluso, realizar una reducción de espacio. El punto de vista del trabajo para esta primera fase se supone funcional; es por ello que, en esta etapa, no se ha visto el trabajo como un proyecto acabado, si no como un proyecto a funcionar de la mejor manera posible. Por otro lado, se puede comprobar que el sensor TCS3200, que medía el color de la hoja, ha sido recubierto por un encapsulado de color negro (Figura 49). Esto se ha realizado para evitar que las condiciones de luz ambiental afecten en la medida del color, siendo ésta, pues, más precisa.

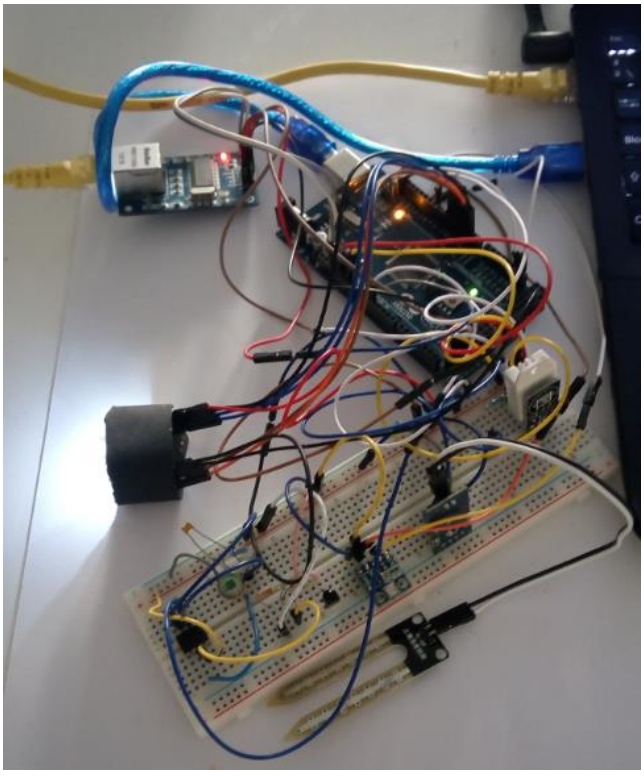


Figura 48. Estación de sensores completa (Arduino, placa con sensores y ENC28j60)

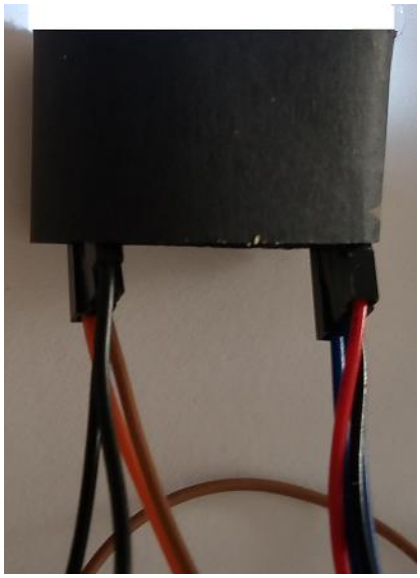


Figura 49 Protección TCS3200 frente a luz ambiental

Como podemos ver en la figura 48, la alimentación del sensor se va a realizar mediante el uso del cable USB (cable azul) que conectará con el ordenador y, además, la subida de las medidas a la base de datos se hará mediante el nuevo módulo que podemos ver en la misma figura (estación de sensores en su conjunto), teniendo una especial visualización del mismo en la Figura 50. Se trata del módulo de Ethernet ENC28j60 que, aunque se explicará en el siguiente capítulo más profundamente, se adelanta acerca de que su funcionamiento consistirá en subir las medidas realizadas, mediante el uso de "queries" (peticiones del software a la base de datos" a la base de datos utilizada). Como bien indica su nombre, utilizará el cable de Ethernet común conectado a router. Se hace saber que este proyecto se deberá entender como una primera fase de un trabajo que será desarrollado y mejorado en fase posteriores. De todo ello, junto a lo anteriormente mencionado acerca de la reducción de espacio y mejora en el orden, hablaremos en el apartado de "Futuras líneas de trabajo" donde se explicará otro método de atención diferente al ordenador, además de otra táctica para enviar datos en modo "wireless".

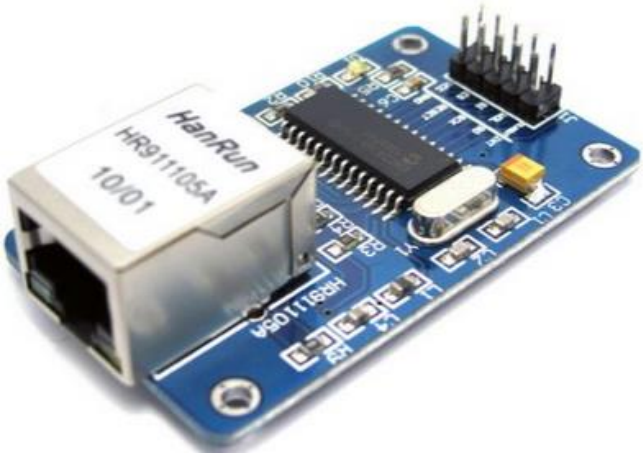


Figura 50. Módulo Ethernet ENC28j60

Para finalizar este capítulo, se procede a mostrar en la Figura 51 el resultado de salida final (de forma informativa, ya que no se precisará del puerto serie para este proyecto, aunque sí como forma útil para comprobar el correcto funcionamiento) del programa de Arduino teniendo en cuenta la implementación de todos los sensores. En esta ocasión, la programación referente a los sensores utilizados no se mostrará debido a su longitud y, además, a que ya se ha mostrado una programación individual de cada sensor en el Capítulo 4 de este trabajo, por lo que mostrar una implementación global en la cual solo se ha unido parte del código de cada sensor se podría considerar una repetición innecesaria del mismo. Por otro lado, la programación en lenguaje Arduino necesaria para la subida de las medidas a la base de datos se mostrará y explicará en el siguiente capítulo.

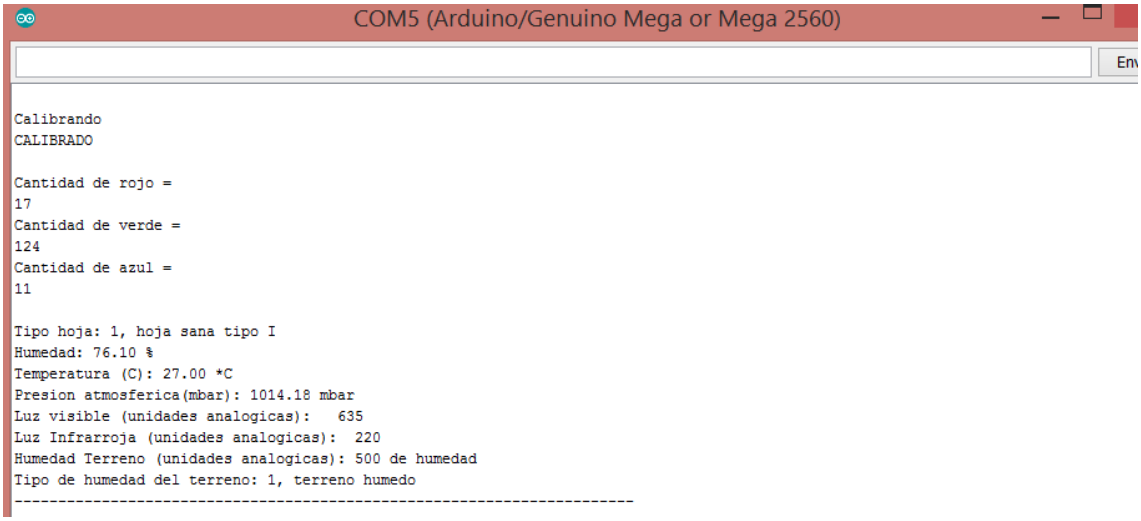


Figura 51. Medidas realizadas desde la estación sensora con Arduino

Se pueden diferenciar tres partes principales. Una parte de calibrado necesaria para el sensor TCS3200 (color de la hoja) como bien se indicó en las especificaciones del sensor. Una segunda parte en la que se especifica la cantidad de rojo, verde y azul (RGB) del color de la hoja medido. Por último, la más importante: una tercera parte donde podemos ver los resultados de las medidas de humedad, temperatura, tipo de hoja analizada, humedad del terreno y tipo, presión atmosférica, luz visible y luz infrarroja. Entre medida y medida se ha establecido un tiempo de 4 segundos para asegurar el correcto funcionamiento de todos los sensores.

Se ha de saber que a la base de datos solo se subirán los datos numéricos de esta última parte, siendo el resto un "adorno" para poder visualizar de mejor manera los datos y entenderlos sin problemas. También hay que tener en cuenta que esta estación sensora es un prototipo para la medida de los parámetros, siendo necesario pensar la implementación de un protector del sistema contra lluvias debido a que éste estará expuesto a la intemperie. Se hablará de esta acción en la sección de "Futuras líneas de trabajo" de este proyecto.

Capítulo 7. Gestión del servidor (Base de datos)

Como bien se ha mencionado en el capitulo anterior, una vez obtenidos las datos resultantes de las medidas realizadas con los sensores utilizados, éstos se deben almacenar para que una aplicación Android pueda cogerlos posteriormente y poder así visualizarlos en la pantalla del terminal.

Se ha decidido que la secuencia a seguir para poder subir dichos datos y almacenarlos sea la siguiente. En primer lugar, la utilización del módulo ENC28j60 que conllevará un conexionado y, como era de esperar, una programación en lenguaje Arduino para poder enviar los datos mediante este módulo. Seguidamente, la utilización del servidor XAMPP que se encargará de la gestión de bases de datos MySQL, del servidor web Apache y de los intérpretes para lenguajes de script, que en nuestro caso utilizaremos PHP. Se hace saber que se utilizará la herramienta phpMyAdmin para manejar la administración de MySQL a través del localhost, utilizando el navegador (localhost/phpmyadmin), donde se creará la base de datos, la tabla de almacenamiento y se definirá un usuario específico de acceso. Por último, se necesitará programar los archivos .php necesarios para poder subir datos. Todas estas fases se explican a continuación detalladamente.

En la Figura 52 se observa el esquemático de engarce del módulo ENC28j60 con Arduino Mega, donde podemos ver que se trata de una simple conexión de 6 pines de dicho módulo.

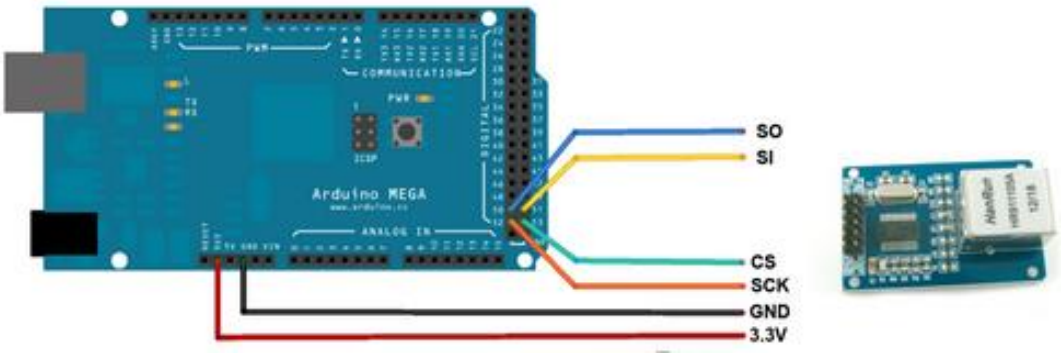


Figura 52. Conexión ENC28j60 con Arduino Mega

Para poder utilizar este módulo en Arduino y subir los datos obtenidos se han de importar varias librearías: “Dhcp.h”, “Dns.h”, “Ethernet_comp.h”, “UIPClient.h”, “UIPEthernet.h” “UIPServer.h” y “UIPUdp.h”. Todas son originarias del Arduino Uno. Para el caso del Arduino Mega, por suerte, se mantienen la gran mayoría, siendo “UIPEthernet.h” la única librería a modificar ya que es ésta la que tiene la correspondencia de los pines. En esta librería, en la parte de la definición de los pines, se ha de cambiar el valor de 4 variables, como se puede ver en la tabla 5.

ENC28J60	Arduino UNO	Arduino MEGA / DUE
MISO (SO)	MISO (pin 12)	MISO (pin 50)
MOSI (SI)	MOSI (pin 11)	MOSI (pin 51)
SCK	SCK (pin 13)	SCK (pin 52)
$\overline{\text{RESET}}$	$\overline{\text{RESET}}$	$\overline{\text{RESET}}$
$\overline{\text{INT}}$	INT0 (pin 2)	INT4 (pin 2)
$\overline{\text{SS}}$ ($\overline{\text{CS}}$)	$\overline{\text{SS}}$ (pin 10)*	$\overline{\text{SS}}$ (pin 53)*
Vdd (3,3 V)	3V3	3V3
Vss (GND)	GND	GND

Tabla 5. Correspondencia de conexión del ENC28j60 con Arduino

Deberemos cambiar, pues, el valor de MISO de 12 a 50, de MOSI de 11 a 51, de SCK de 13 a 52 y de SS de 10 a 53. Esto es para darle a entender que se está usando Arduino Mega y que esos son los pines que se van a utilizar.

Una vez conectado el ENC28j60 y habiendo modificado las librerías, se procede a mostrar la programación en lenguaje Arduino empleado para poder realizar la conexión con la base de datos y, seguidamente, realizar los envíos de los mismos. Esta programación la podemos ver en las Figuras 53, 54, 55 y 56.

```
#include <UIPClient.h>
#include <UIPEthernet.h>
#include <UIPServer.h>
#include <UIPUdp.h>
EthernetClient client;
byte ip[] = {192,168,1,206};
```

Figura 53. Librerías importadas, creación del objeto cliente y definición de la IP

En la figura anterior se puede observar la importación de las librearías anteriormente mencionadas y la creación del objeto cliente de la clase EthernetClient, para poder usar sus métodos más adelante. También, la definición de nuestra IP para poder ejecutar la conexión.

```
void setup()
{

  TSC_Init();
  Serial.begin(9600);
  uint8_t mac[6] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05};
  Ethernet.begin(mac,ip);
```

Figura 54. Definición de una MAC aleatoria (sin ocupar) y el inicio de la definición de la conexión

En el setup se establece los parámetros de la conexión, pasándole una MAC, que nos podemos inventar siempre y cuando no esté ocupada por ningún otro terminal y siguiendo el formato mostrado en la imagen, y la IP local de tu terminal necesaria para poder realizar la conexión.

```
  if (client.connect(IPAddress(192, 168, 1, 206), 80)) {
{
  client.print("POST /sergio/subir_datos.php?");
  //Inicio del query, sensor 1
  client.print("s1=");
  client.print(h);  //humedad
  //Sensor 2
  client.print("&&");
  client.print("s2=");
  client.print(t); //temperatura
  //Sensor 3
  client.print("&&");
  client.print("s3=");
  client.print(mbar); //presión
  //Sensor 4
  client.print("&&");
  client.print("s4=");
  client.print(miLV);  //Luz visible
  // Sensor 5
  client.print("&&");
  client.print("s5=");
  client.print(miIR); //Luz infrarroja
  // Sensor 6
  client.print("&&");
  client.print("s6=");
  client.print(tipoHoja); //humedad terreno
  //Sensor 7
  client.print("&&");
  client.print("s7=");
  client.print(humTersal); //tipo humedad terreno
  //Fin del query
  client.println(" HTTP/1.0");
  client.println("User-Agent: Arduino 1.0");
  client.println();

  Serial.println(">> Dato enviado");
```

Figura 55. Realización de la Query y envío de datos

En esta figura anterior se puede observar la realización de la query necesaria para que se produzca el envío de datos. En primer lugar, se observa un condicional "if" en el que se comprueba si estamos conectados a la base de datos para poder realizar la petición. Si estamos conectamos, se produce la definición de la "query" realizada por POST y en la que llamaremos al archivo "subir_datos.php" para pasarle los datos en cuestión y que sean subidos (se observa el símbolo "?" en la query, necesario para pasar los datos en la petición). Como en nuestros archivos .php tenemos definido que los datos se almacenen en variables llamadas s1, s2, s3, s4, s5, s6 y s7, así lo haremos aquí también. Como se puede ver en la figura 55, cada dato se va metiendo en una variable y se va concatenando con "&&" en la query. Finalmente, una vez formada la query (?s1=a&&s2=b…), se procede al envío de los datos. Se ha añadido un chivato al final (">>Dato enviado") para indicar que hemos entrado a esa condición y que, efectivamente, se ha formado correctamente la query. El envío de datos se producirá gracias al archivo "subir_datos.php".

```
} else {
  Serial.println("* Fallo en la conexion *");

}
if (!client.connected()) {
  Serial.println("<< Desconectado");
}
client.stop();
client.flush();
 delay (1000);
```

Figura 56. Mensajes en caso de error y al desconectar el cliente

Si por un casual no se conectara bien a la base de datos o se desconectara el cliente, mostraría los mensajes que se observan en la Figura 56 para indicarlo. Por otro lado, después del envío de datos el sistema se para y, mediante el uso de "client.flush", se espera hasta que se hayan enviado todos los caracteres salientes del buffer.

Una vez vista la programación, se continuará con la explicación de la conexión a base de datos usando XAMPP y phpMyAdmin, así como la creación de la misma y una tabla de almacenamiento de valores. Repitiendo lo dicho anteriormente, se va a utilizar el servidor XAMPP, un servidor independiente de plataforma de código libre que te permite instalar de forma sencilla Apache en tu propio ordenador, sin importar tu sistema operativo. Entre otras cosas, incluye servidores de base de datos como MySQL con sus respectivos gestores, en nuestro caso, phpMyAdmin.

Una vez instalado XAMPP, en el buscador de programas localizaremos "XAMPP Control Panel" y abriremos dicho programa, teniendo lo observado en la Figura 57.

Figura 57. XAMPP Control Panel (servidores inactivos)

Una vez abierta la pantalla de la Figura anterior, tendremos que poner en marcha el servidor Apache y, también, el servidor de base de datos MySQL. Para ello, para ambos casos, pulsaremos Start, teniendo un resultado como el que se puede ver en la Figura 58.

Figura 58. XAMPP Control Panel (Servidores activos)

Ahora ya están arrancados los servidores, por lo tanto, ya nos podemos conectar a la herramienta de gestión phpMyAdmin. Como lo vamos a realizar en local, en la barra de dirección del explorador que utilices (Google Chrome, IE u otro), escribiremos "localhost/phpmyadmin" para proceder a realizar la base de datos. En la Figura 59 podemos ver el resultado de introducir dicha URL.

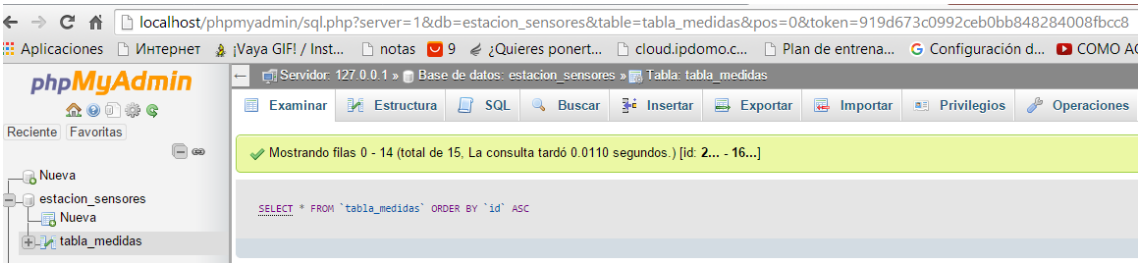


Figura 59. Gestión de la base de datos

Se ha elegido phpMyAdmin para gestionar la base de datos por su facilidad de uso y entendimiento. Para crear una nueva base de datos sólo hay que seleccionar la palabra "Nueva" que vemos en la figura anterior y darle un nombre. Por otro lado, para crear una nueva tabla dentro de dicha base de datos, entraremos dentro de la misma y pulsaremos el botón de nueva tabla junto al número de variables que queremos registrar. Hay que tener en cuenta que siempre habrá que poner 2 variables más de las que tenemos, es decir, si tuviéramos 7 variables (de s1 a s7) como en nuestro caso, el número de variables a poner al inicializar la tabla sería 9 debido a que es necesario introducir al principio un identificador (Id autoincrementable) y una variable temporal para controla en qué día y momento del mismo se registraron dichos datos. En esta ocasión, la base de datos se llama "estación_sensores" y la tabla "tabla_medida".

Con la base de datos creada y la programación en Arduino realizada, solo queda por mostrar los archivos .php necesarios para poder realizar la conexión y subir los datos; estos archivos los podemos ver en las Figuras 60 y 61.

```php
<?php
    // config.php
    // Credenciales
    $dbhost = "localhost";
    $dbuser = "admin";
    $dbpass = "admin";
    $dbname = "estacion_sensores";
    // Conexión con la base de datos
    $con = mysqli_connect($dbhost, $dbuser, $dbpass, $dbname);
?>
```

Figura 60. Archivo config.php

Este primer archivo llamado "config.php" se usa para definir cuál será el host (localhost), el nombre de la base de datos y la acreditación. Esta acreditación consistirá en una autenticación basada en usuario y contraseña. Estos "roles" de acceso a la base de datos se pueden establecer en phpMyAdmin seleccionando la base de datos y, luego, pulsando "cuentas de usuario"; ahí podemos añadir usuarios y darle (o no) privilegios. Por último, gracias a este archivo, se realiza la conexión con la base de datos.

```php
<?php
    // Importamos la configuración
    require("config.php");

    // Esta es la instrucción para insertar los valores en la tabla
    $query = "INSERT INTO tabla_medidas(s1,s2,s3,s4,s5,s6,s7) VALUES ('".$_GET["s1"]."', '".$_GET["s2"]."',
    '".$_GET["s3"]."','".$_GET["s4"]."','".$_GET["s5"]."','".$_GET["s6"]."','".$_GET["s7"]."')";

    // Ejecutamos la instrucción
    mysqli_query($con, $query);
    mysqli_close($con);
?>
```

Figura 61. Archivo subir_datos.php

Este último archivo, "subir_datos.php" importa la configuración (para realizar la conexión con la base de datos) e inserta los valores en la tabla mediante la query.

Conocidos los archivos .php necesarios para conectar a la base de datos, lo único que queda es ejecutar el programa Arduino que montará los valores recogidos de los sensores en la query del archivo "subir_datos.php" y, mandando dicha petición, los subirá a la base de datos, tal y como se puede observar en la Figura 62, donde podemos ver la base de datos con valores ya registrados al realizar los pasos anteriormente mencionados.

	id	s1	s2	s3	s4	s5	s6	s7	tiempo
Editar Copiar Borrar	2	50.3	24.3	1014.23	635	207	1	2	2016-08-31 20:09:44
Editar Copiar Borrar	3	50.2	24.3	1014.23	635	207	1	2	2016-08-31 20:09:50
Editar Copiar Borrar	4	50.3	24.3	1014.25	634	207	1	4	2016-08-31 20:09:56
Editar Copiar Borrar	5	50.3	24.3	1014.25	620	207	1	4	2016-08-31 20:10:02
Editar Copiar Borrar	6	50.3	24.4	1014.25	622	207	1	4	2016-08-31 20:10:08
Editar Copiar Borrar	7	50.3	24.3	1014.25	625	315	1	3	2016-08-31 20:10:14
Editar Copiar Borrar	8	50.3	24.4	1014.25	630	212	5	3	2016-08-31 20:10:20
Editar Copiar Borrar	9	50.3	24.3	1014.25	630	212	5	3	2016-08-31 20:10:26
Editar Copiar Borrar	10	50.3	24.3	1014.25	630	212	5	3	2016-08-31 20:10:32
Editar Copiar Borrar	11	50.3	24.3	1014.25	631	211	5	3	2016-08-31 20:10:38
Editar Copiar Borrar	12	50.3	24.3	1014.25	631	211	5	3	2016-08-31 20:10:44
Editar Copiar Borrar	13	50.3	24.3	1014.25	630	211	5	3	2016-08-31 20:10:50
Editar Copiar Borrar	14	50.3	24.3	1014.25	630	210	5	3	2016-08-31 20:10:56
Editar Copiar Borrar	15	50.3	24.3	1014.25	630	210	5	3	2016-08-31 20:11:02
Editar Copiar Borrar	16	50.3	24.3	1014.25	630	210	5	3	2016-08-31 20:11:08

Figura 62. Tabla de la base de datos con valores registrados

Como bien se comentó unos párrafos más arriba, se observa una tabla con 9 columnas (id, 7 variables a registrar y la fecha y la hora cuando se realizó la medida). Como se puede comprobar, siguiendo los pasos anteriormente mencionados, ya tenemos montada y registrando la base de datos por lo que se va a comenzar la siguiente tarea: programación de la aplicación que se conectará con la base de datos para obtener los valores de las medidas realizadas.

Capítulo 8. Aplicación Android para móvil

En último lugar, se procede a realizar una aplicación Android para el móvil, dando así por completado la realización del trabajo. Como bien se ha comentado anteriormente, una vez montada toda la estación sensora y, posteriormente, conectada a una base de datos para la subida de los mismos a tiempo real, solo quedaría programar un método de recogida de dichos datos siguiendo los términos de "whenever, anywhere", es decir, dónde sea y cómo sea siempre y cuando se disponga de conexión a internet.

Para este proyecto se ha decidido realizar la aplicación con App Inventor 2, una plataforma de Google Labs para crear aplicaciones compatibles con el sistema operativo Android. Su programación se basa en un concepto visual mediante la unión de diferentes bloques funcionales. Se ha decidido este sistema debido a que los medios de los que se disponían (ordenador para programar) no eran aptos para programación en Android Studio ni Eclipse.

Se ha determinado comenzar primero con la interfaz de la aplicación (la parte visual) y después, una vez realizada todas las pantallas que componen la aplicación, dotarlas de funcionalidad y lógica. En primer lugar se ha construido la pantalla de inicio, observable en la Figura 63, la cual está compuesta por un botón de cierre de la aplicación en forma de equis roja, el título de la aplicación y cuatro botones que nos llevarán a una pantalla distinta cada uno.

Figura 63. Pantalla Inicio aplicación Android

Se hace saber que los rectángulos negros, que aparecen en la figura anterior y en las posteriores, realmente no se visualizarán en la actividad de la aplicación. Solo son posicionadores de elementos a lo largo de la pantalla, pero al ejecutar la app desaparecen.

Por no alargar mucho la presentación gráfica de la aplicación, se procede a visualizar de forma continua todas las pantallas de la misma en las Figuras 64 y 65:

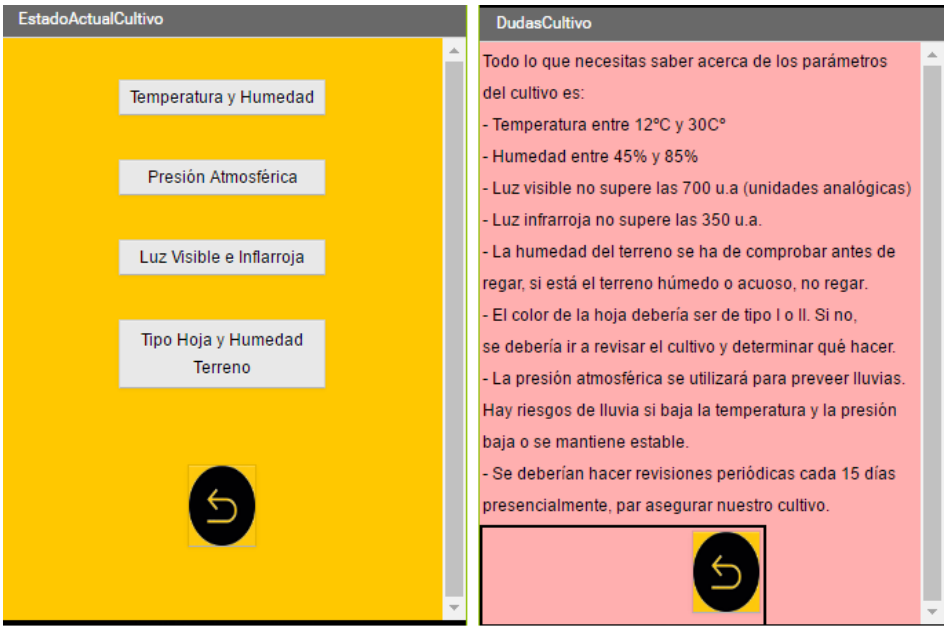


Figura 64. Pantalla del estado actual del cultivo (izquierda) y de las dudas posibles (derecha)

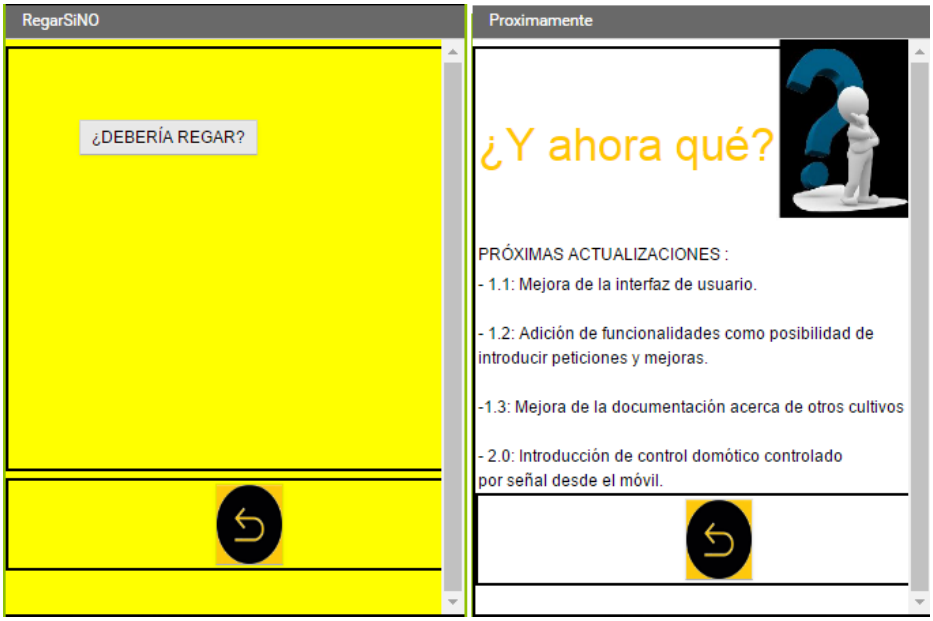


Figura 65. Pantalla para decidir si realizar el regado (izquierda) y de las futuras actualizaciones (derecha)

Explicando brevemente sobre la función de cada una de las pantallas anteriores, la pantalla Próximamente nos ayuda a saber cuáles serán las próximas actualizaciones que tendrá esta aplicación. Se piensa que es necesaria debido a que así se puede informar a los usuarios de próximas funcionalidades de la misma, así como permitir que propongan ideas a desarrollar.

Por otro lado, el funcionamiento de la pantalla RegarSiNO se basa en, como indica el nombre, aconsejarnos acerca de si deberíamos regar en ese momento o no. Aunque más adelante se explicará la lógica empleada, se adelanta que recogerá algunos parámetros y, según sus valores, se decidirá si se debería regar o no, ayudando así al usuario. Al darle Al botón de "¿DEBERÍA REGAR?" nos responderá con un "SÍ/NO", siendo esta última opción justificada mediante razones.

DudasCultivo es otra pantalla que, desde el punto de vista de usuarios iniciales en el mundo de la agricultura, puede ser muy útil. Esto es debido a que en esta parte de la aplicación se desglosa una información necesaria para atender los cultivos y saber cuál es su estado según los valores obtenidos. Es meramente informativa.

En último lugar, la pantalla de EstadoActualCultivo nos permite, mediante otros 4 botones, acceder a 4 pantallas de monitorización de los parámetros. Estas 4 nuevas pantallas se pueden observar en las Figuras 66 y 67:

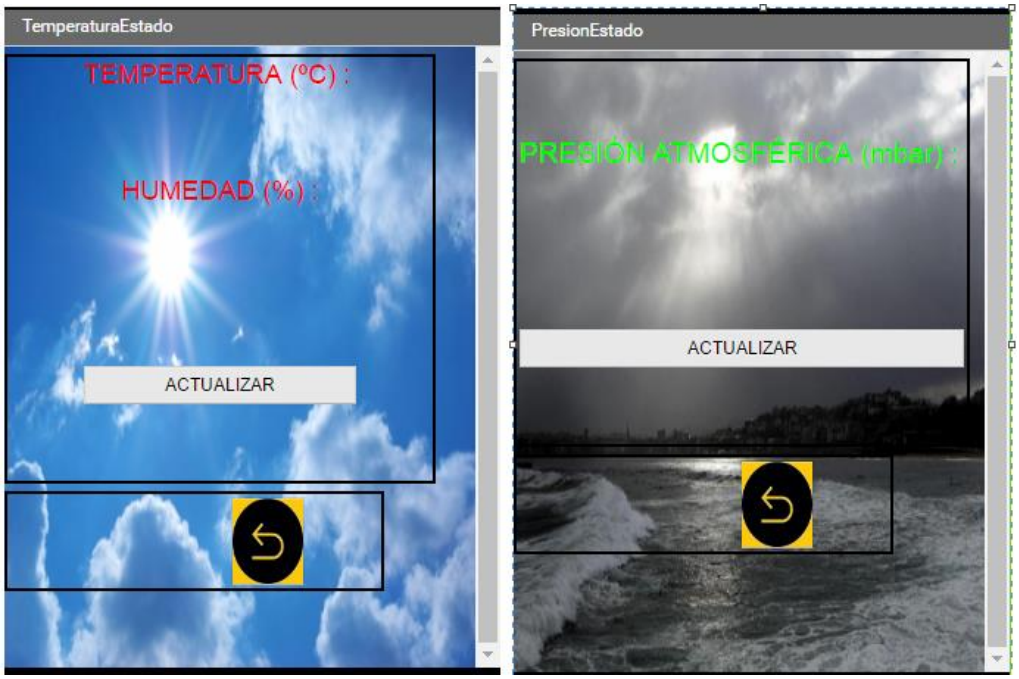


Figura 66. Pantalla para monitorizar la temperatura y la humedad (izquierda) y la presión atmosférica (derecha)

Figura 67. Pantalla para monitorizar la cantidad de luz visible e infrarroja (izquierda) y el tipo de hoja y humedad del terreno (derecha)

En todas las pantallas anteriores, pulsando el botón de actualizar, podremos saber en cada momento en qué condiciones de temperatura, humedad, etc., se encuentra nuestro cultivo. Por último, se hace entender que el botón de la flecha amarilla/negra es para permitir al usuario moverse entre pantallas con facilidad, permitiendo el mismo volver hacia la pantalla anterior.

Una vez mostrada la interfaz gráfica usada para este proyecto, se continua con la implementación de la lógica y la funcionalidad de cada una de las pantallas (esto es, entre otras cosas, permitir moverse entre pantallas, conectarse a base de datos, etc.).

En la mayoría de pantallas, tales como la pantalla principal (Screen1), Próximamente, EstadoActualCultivo y DudasCultivo, la única lógica/funcionalidad que se ha implementado es la de cambiar de pantalla utilizando los botones disponibles para ello, ya sean los de la flecha hacia atrás o los propios botones de elección. En App Inventor 2 se ha utilizado el bloque mostrado en la Figura 68 para poder dotarla de dichas acciones.

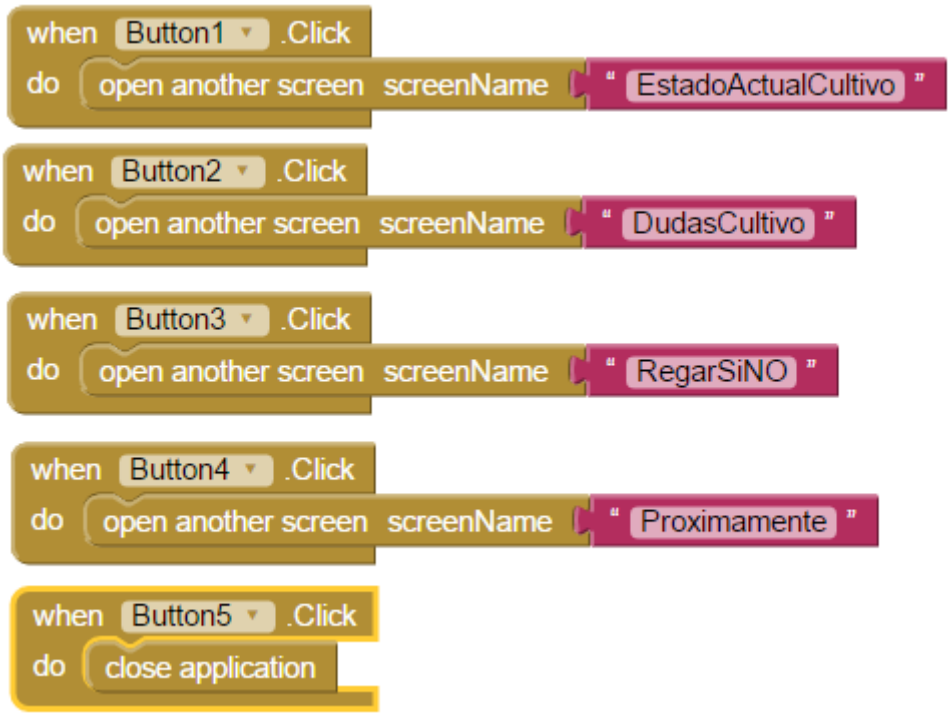


Figura 68. Bloques necesarios para movimiento entre pantallas

Como se puede ver, se basa simplemente en una condición la cual, una vez pulsado el botón en cuestión, abre otra pantalla con el nombre indicado. Otra de las opciones es que, al pulsarlo, se cierre la aplicación directamente.

En el resto de pantallas, en las cuales se necesitan obtener datos de la base de datos, se ha utilizado el bloque mostrado en la Figura 69 donde podemos ver que, de nuevo, al pulsar el botón (Actualizar) llama a un proceso que se encarga de mostrar en forma de texto ("label") las valores recogidos de la base de datos, además de almacenarlos localmente para poder utilizar esos datos entre pantallas (sin la llamada a TinyDB1 sería imposible utilizar datos entre funcionalidades de diferentes pantallas) mediante el uso de "tags".

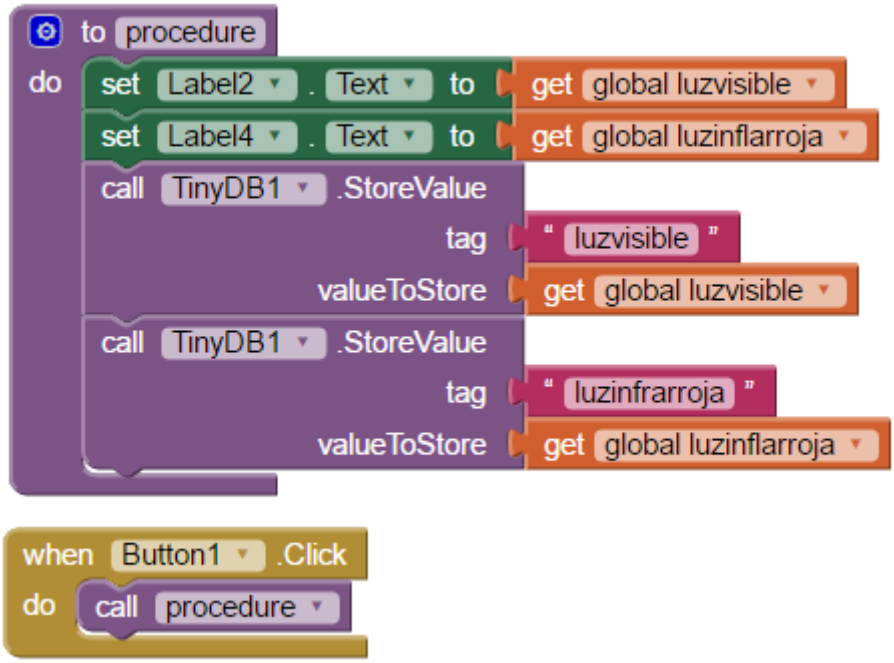


Figura 69. Bloques necesarios poder mostrar los datos por pantalla

Se muestra también, para completar este capítulo, la realización de la conexión con la base de datos generada anteriormente en phpMyAdmin. Para ello se han utilizado los bloques que podemos ver en las Figuras 70 y 71.

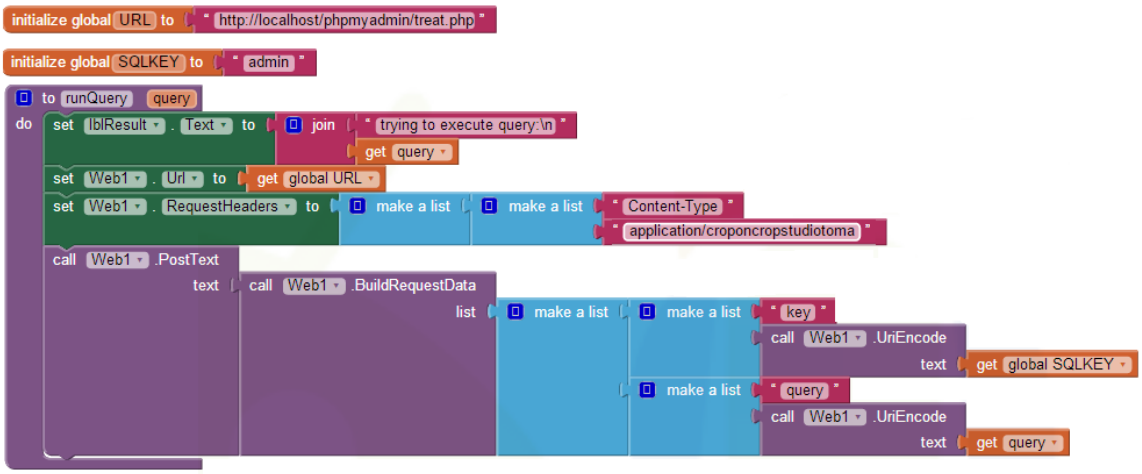


Figura 70. Bloques necesarios conexión con base de datos

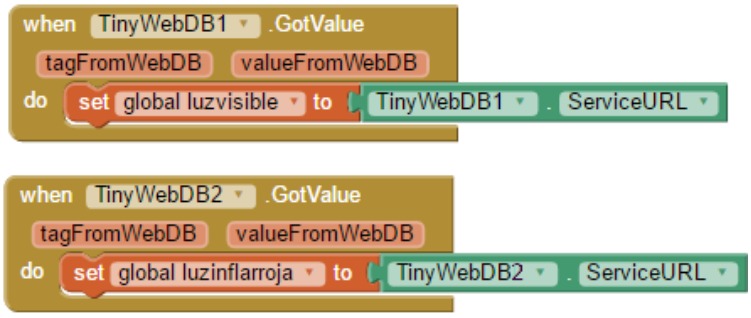


Figura 71. Bloques necesarios poder obtener datos mediante un WebServices apoyado por TinyWebDB1

Se basa en la realización de una query en la cual se pide que se conecte a la base de datos y ejecute un archivo .php al que se accede mediante la URL. Luego, como se ve en la figura anterior, una vez ejecutado el .php que implementa el comportamiento lector de la aplicación y pasándole una query similar a la de la Figura 72, se accede a los datos que podremos almacenar en variables globales, estando ya disponibles para su uso.

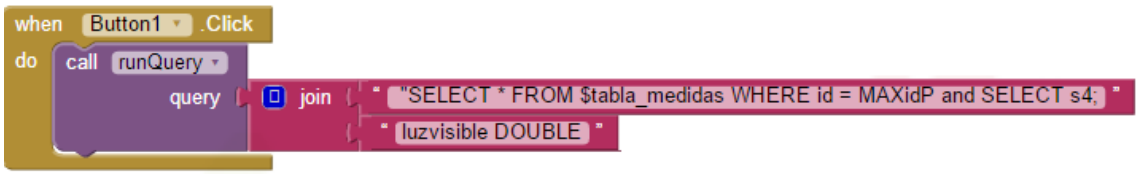


Figura 72. Bloques necesarios para definir la petición a la base de datos

Este código .php que podemos ver en la Figura 73 y la Figura 74 ejecuta una petición de lectura y, además, indica que esa lectura sea con respecto al "id" más alto, haciendo así que coja el valor más actual de las medidas.

```
<?php

/************************************CONFIG*****************************************/
//DATABSE DETAILS//
$DB_ADDRESS="http://192.168.1.206/";
$DB_USER="admin";
$DB_PASS="admin";
$DB_NAME="estacion_sensores";

$SQLKEY="secret";

/************************************CONFIG*****************************************/


header('Cache-Control: no-cache, must-revalidate');

error_log(print_r($_POST,TRUE));

if( isset($_POST['query']) && isset($_POST['key']) ){
  header('Content-type: text/csv');

  if($_POST['key']==$SQLKEY){                                          //validates the SQL key
    $query=urldecode($_POST['query']);
    if(get_magic_quotes_gpc()){
      $query=stripslashes($query);
    }
    $conn = new mysqli($DB_ADDRESS,$DB_USER,$DB_PASS,$DB_NAME);    //connect

    if($conn->connect_error){                                          //checks connection
      header("HTTP/1.0 400 Bad Request");
      echo "ERROR Database Connection Failed: " . $conn->connect_error, E_USER_ERROR;  //reports a DB connection failure
    } else {
      $result=$conn->query($query);                                    //runs the posted query
      if($result === false){
        header("HTTP/1.0 400 Bad Request");                            //sends back a bad request error
        echo "Wrong SQL: " . $query . " Error: " . $conn->error, E_USER_ERROR;  //errors if the query is bad and spits the error back to the client
      } else {
        if (strlen(stristr($query,"SELECT"))>0 && MAXidP) {                    //tests if it's a SELECT statement
          $csv = '';                                                   // bug fix Undefined variable: csv
          while ($fieldinfo = $result->fetch_field()) {
            $csv .= $fieldinfo->name.",";
          }
          $csv = rtrim($csv, ",")."\n";
          echo $csv;                                                   //prints header row
          $csv = '';

          $result->data_seek(0);
          while($row = $result->fetch_assoc()){
            foreach ($row as $key => $value) {
              $csv .= $value.",";
            }
            $csv = rtrim($csv, ",")."\n";
          }
          echo $csv;                                                   //prints all data rows
```

Figura 73. Código php necesario para poder realizar la petición usando la query (parte I)

```
        } else {
          header("HTTP/1.0 201 Rows");

        }
      }
      $conn->close();                                    //closes the DB
    }
  } else {
     header("HTTP/1.0 400 Bad Request");
     echo "Bad Request";                                 //reports if the secret key was bad
  }
} else {
       header("HTTP/1.0 400 Bad Request");
       echo "Bad Request";
}
```

Figura 74. Código php necesario para poder realizar la petición usando la query (parte II)

Después de esto, ya se tiene implementada en su totalidad la aplicación en cuestión, dotada de la funcionalidad suficiente para lo que este proyecto busca realizar. Se procede, pues, a mostrar algunas de las funcionalidades de la app que se pueden ver en la Figura 75 y 76.

Figura 75. Funcionalidad de decisión de regado

Esta pantalla de la app ha sido dotada con una comprobación de los parámetros y un campo de decisión que te recomendará si debes regar o si no, así como la justificación de su decisión mediante el uso de “Razones”.

Figura 76. Funcionalidad de monitorización de parámetros a tiempo real

Por último, en todas las pantallas referentes a la medida de parámetros que afectan al crecimiento de la tomatera, se ha dotado al botón de actualizar de una funcionalidad de "pass/notpass", es decir, sólo mostrará los datos una vez hayas pulsado el botón. Esto es útil para lo relacionado con no consumir recursos innecesarios durante la ejecución de la aplicación. En el momento del pulsado, se conecta a la base de datos y recoge el último valor almacenado en la columna correspondiente.

Con este último paso se concluye finalmente la implementación de la aplicación y, con ello, la construcción total del sistema buscado. Por hacer un breve resumen de todo, se ha construido una estación con sensores que, mediante programación Arduino, y el acondicionamiento necesario (o no), es capaz de registrar datos con respecto al ambiente que rodea a un producto concreto, la tomatera, siendo estos datos registrados la temperatura, la humedad, presión atmosférica, etc.

Por otro lado, se ha implementado una base de datos que, conectando el Arduino con la misma, se ha podido almacenar todos los datos a tiempo real para que luego pudieran ser accesibles mediante la aplicación Android que acabamos de programar.

Se finalizan, pues, los capítulos de implementación dejando paso así a temas menos técnicos como son los presupuestos y planos y las conclusiones, entre otros.

Capítulo 9. Pliego de condiciones

9.1 Plano de un cultivo real

Se procede a explicar cual sería la distribución óptima para la parametrización de más de una planta tomatera, es decir, se habla ya de términos más grandes tal como es un huerto completo de tomates. En el caso de disponer de un cultivo extenso de tomateras, no es posible colocar una única estación sensora debido a la imprecisión que puede suponer dicha acción. Esto es debido a que una sola muestra tomada de una gran población puede suponer que no acierte en lo que el cultivo en general se refiere, pudiéndose dar el caso de enfermedad de muchas y, casualmente, no enfermedad de la muestra en cuestión medida.

Antes de todo hay que saber que, normalmente, los cultivos están distribuidos en diferentes franjas, es decir, diferentes "líneas" excavadas en el terreno donde se va a introducir el cultivo. Si bien no es obligatorio, se aconseja que como muy mínimo se tenga una estación sensora por zanja o línea de cultivo. Aunque, desde el punto de vista de la precisión, estas medidas podrían de nuevo quedarse cortas. Es por ello que se procede a mostrar, en la Figura 77, cual sería la distribución más óptima para parametrizar dichos valores, teniendo en cuenta también lo que supondría económicamente.

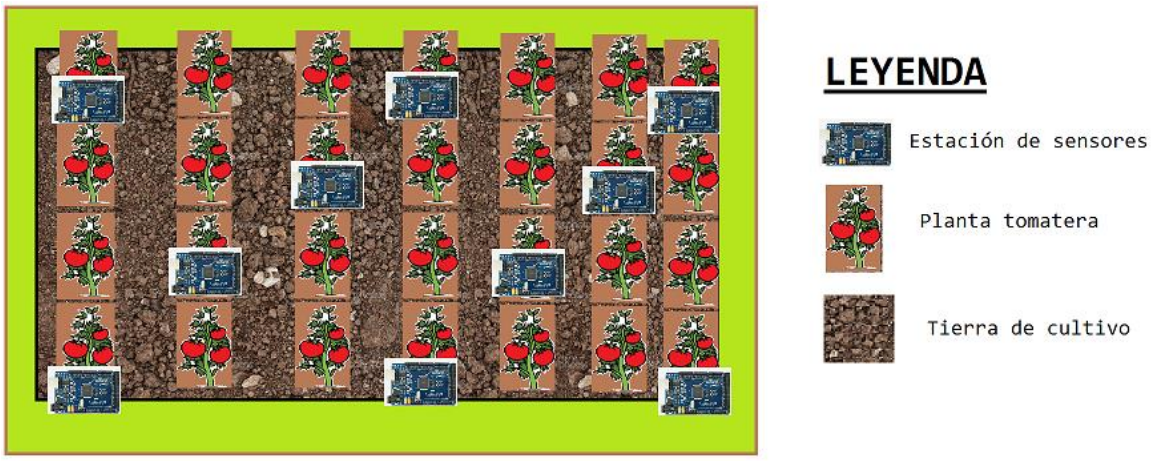


Figura 77. Distribución óptima de las estaciones de sensores en un cultivo real

Como podemos comprobar, existen 4 zanjas con 7 plantas tomateras cada una, con lo que en total tenemos unas 28 plantas tomateras. La regla seguida para la distribución de cada estación de sensores ha sido, comenzando por la primera zanja, colocar la primera estación de sensores en la primera tomatera y, desde ahí, ir colocando cada 3 plantas tomateras otra estación de izquierdas a derechas. Esto se ha hecho para obtener con más alta probabilidad, y no abusando de la parte económica, valores más fiables (aunque no definitivos, es obvio que no será preciso al 100% si no compruebas el 100% de las muestras) al tener como mínimo 2 estaciones en cada zanja de 7 tomateras. Si bien se ha elegido éste como el modelo más óptimo en relación calidad-precio (10 estaciones para 28 plantas) se podrá disponer, según gustos y dinero disponible, de otras distribuciones diferentes. Obviamente, si tenemos una estación sensora en cada planta tomatera debido a buena solvencia económica, la exactitud en las medidas aumentaría al tener bajo parametrización a todas las muestras de un cultivo.

Más adelante, en el apartado de "Futuras líneas de trabajo", se hablará de las posibles mejoras de implementación de estos sensores. Se ha de recordar que este trabajo no estaba pensado inicialmente para controlar cultivos gigantes, tales como los de grandes empresas, si no para ayudar al pequeño agricultor que no pueda estar pendiente de su huerta presencialmente debido

a enfermedad u otros, o en el caso de que una persona sin conocimientos necesarios se tuviera que hacer cargo de dicha huerta.

9.2 Presupuesto

En este apartado del capítulo 9 se va a realizar un presupuesto individual del proyecto general de la estación sensora. Se da por hecho que el listado de materiales se incluirá en este presupuesto, junto al precio individual por pieza, así como un precio total del sistema completo. Además, se han considerado las horas de estudio y desarrollo del sistema, aunque sólo se tendrán en cuenta para esta primera implementación, ya que una vez estudiado y diseñado una estación, el resto sigue el mismo procedimiento. El presupuesto se presenta en la Figura 78, realizado en una hoja de Excel, donde se ha tenido en cuenta los precios reales de compra que supuso para la realización de dicho trabajo.

PRESUPUESTO			
MATERIAL	UNIDADES	PRECIO UNITARIO	PRECIO TOTAL
Arduino Mega 2560	1	11,75 €	11,75 €
ENC28j60 Ethernet LAN Network Module	1	7,40 €	7,40 €
Cables Ethernet RJ-45 (20 metros)	2	6,75 €	13,50 €
Conjunto 65 cables Arduino	1	1,71 €	1,71 €
Sensor BMP180	1	5,45 €	5,45 €
Sensor BPW21R	1	9,85 €	9,85 €
Sensor BPV22F	1	4,43 €	4,43 €
Sensor DHT22	1	3,20 €	3,20 €
Sensor TCS3200	1	4,70 €	4,70 €
Conjunto sensor YL-69 e YL-38	1	4,15 €	4,15 €
TL081CN	2	0,99 €	1,98 €
Condensador 10 pF	2	0,68 €	1,36 €
Resistencia 4,7 kΩ	2	0,04 €	0,08 €
Resistencia 12 kΩ	2	0,04 €	0,08 €
Resistencia 200 kΩ	2	0,04 €	0,08 €
Resistencia 10 kΩ	1	0,04 €	0,04 €
Horas de desarrollo	300	9,00 €	2.700,00 €
		PRECIO FINAL (I.V.A. INCLUIDO)	2.769,76 €

Figura 78. Presupuesto total del sistema

Este precio, como se puede comprobar, está ajustado para la realización de un solo sistema de registro de parámetros. Para el caso expuesto como el óptimo en la Figura 77 con 10 módulos, el resultado rápido sería pensar que el presupuesto ascendería al que se puede observar en la ecuación 9.2.1 (sin tener en cuenta ya el precio por horas de desarrollo)

$$Precio\ 10\ módulos = Precio\ 1\ módulo * 28 = 69.76 * 10 = 697.76\ € \qquad \textbf{(9.2.1)}$$

A pesar de que, si se dispone del dinero, se podría realizar así, se propone un caso alterno en el cual se podrían reutilizar los Arduinos Mega debido a la gran cantidad de pines de entrada/salida de los que disponen. Por lo que, con un cableado más largo nos podríamos ahorrar muchas unidades, debido a que de las 15 entradas que hay analógicas solo usamos 3 (podríamos agrupar hasta 5 módulos) y de las 54 que hay digitales, solo usamos 6 más 4 para el módulo de Ethernet (este módulo de Ethernet ya no será necesario debido a que se propondrá en "Futuras líneas de trabajo" el uso de "wireless") por lo que podríamos agrupar más de 5 plantas. El más restrictivo de los dos es el caso analógico por lo que, teniéndolo en cuenta, cada Arduino podría controlar hasta 5 plantas diferentes (de nuevo, usando cableado más largo para los sensores). Por otro lado, no haría falta un sensor de temperatura y humedad por módulo, debido a que esta se mantendrá a lo largo del terreno. También ocurre lo mismo con el sensor de presión atmosférica, luz visible y luz infrarroja. Se podrían poner 2, o a lo sumo 3, sensores de estos tipos para hacer comprobaciones de correcto funcionamiento a lo largo de todo el cultivo

propuesto en el caso óptimo. Por el contrario, sí que sería necesario poner un sensor de humedad del terreno y del color de la hoja por sistema sensor. Con todo esto, se realiza un nuevo presupuesto (sin considerar el número de horas de desarrollo; si se quiere tener en cuenta, se sumaría 2.700 €) tomando en cuenta las pautas anteriores que puede ser visto en la Figura 79:

PRESUPUESTO			
MATERIAL	**UNIDADES**	**PRECIO UNITARIO**	**PRECIO TOTAL**
Arduino Mega 2560	2	11,75 €	23,50 €
Cables Ethernet RJ-45 (20 metros)	1	6,75 €	6,75 €
Conjunto 65 cables arduino	2	1,71 €	3,42 €
Sensor BMP180	3	5,45 €	16,35 €
Sensor BPW21R	3	9,85 €	29,55 €
Sensor BPV22F	3	4,43 €	13,29 €
Sensor DHT22	3	3,20 €	9,60 €
Sensor TCS3200	10	4,70 €	47,00 €
Conjunto sensor YL-69 e YL-38	10	4,15 €	41,50 €
TL081CN	6	0,99 €	5,94 €
Condensador 10pF	6	0,68 €	4,08 €
Resistencia 4,7 kΩ	6	0,04 €	0,24 €
Resistencia 12 kΩ	6	0,04 €	0,24 €
Resistencia 200 kΩ	6	0,04 €	0,24 €
Resistencia 10 kΩ	3	0,04 €	0,12 €
		PRECIO FINAL (I.V.A. INCLUIDO)	**201,82 €**

Figura 79. Presupuesto total teniendo en cuenta la distribución óptima

Como se puede ver, el precio ha bajado muchísimo con respecto al primer cálculo y se obtendrá una exactitud parecida. Para finalizar el presupuesto anterior, se ha de observar que se ha quitado el módulo de Ethernet ENC28j60; esto es debido a que no hará falta, será sustituido por un módulo Bluetooth (se hablará en el capítulo de "Futuras líneas de trabajo") por cada estación sensora, es decir, 2 módulos bluetooth que supondría un incremento del presupuesto de 10€, llegando la factura total a 211.82€ ≈ 212€.

Capítulo 10. Conclusiones

En este apartado se pretende exponer la visión general acerca de la realización del trabajo: comprobar si se han cumplido los objetivos inicialmente expuestos, hablar acerca del tiempo empleado finalmente y si se ajusta a lo planificado al principio y, finalmente, hacer un breve comentario acerca de si el creador, uno mismo, se siente satisfecho con lo realizado.

En primer lugar, se procede a comprobar si se han cumplido los objetivos inicialmente expuestos:

- **Entender cuáles son las magnitudes físicas principales que afectan a los cultivos**

A lo largo del Capítulo 4 se ha explicado todas las magnitudes físicas a tener en cuenta a la hora de realizar cualquier cultivo pero, para ser más específicos, se ha centrado el enfoque en una planta, la tomatera. Entre esas magnitudes estaban la temperatura, la humedad, la presión atmosférica, la radiación tanto visible como infrarroja, etc., las cuales se han explicado en la medida de lo posible para su fácil entendimiento.

- **Comprender cómo afectan dichas magnitudes a los cultivos**

Del mismo modo, de nuevo en el Capítulo 4, se explica de forma bastante clara cómo podrían perjudicar las magnitudes a medir a nuestros cultivos si se salen del rango de valores esperado para cada una de ellas.

- **Elegir, de forma lógica, los sensores más óptimos que nos permitan cuantificar dichas magnitudes físicas**

Se ha dedicado todo el Capítulo 5, apartado por apartado, a comentar acerca de los sensores elegidos y las razones que se han tenido en cuenta para seleccionarlos finalmente para su uso en la estación sensora. También, además, se ha comentado la programación necesaria y, en caso de necesitarlo, la aplicación del acondicionamiento del sensor para su correcto funcionamiento.

- **Almacenar todos los datos recogidos desde el momento que se conecte el sistema para tener información actual de nuestro cultivo**

En este caso, a través de su explicación en el capítulo 7, se ha mostrado la creación y posterior utilización de una base de datos mediante phpMyAdmin. Se ha hablado de los diferentes archivos .php necesarios para subir datos desde el Arduino hasta dicha base de datos. Finalmente, se ha mostrado dicha subida de datos dando por hecho el cumplimiento de este punto.

- **Acceder a la información del estado del campo a tiempo real mediante nuestro Smartphone o Tablet**

Finalmente, hemos creado una aplicación la cual se conectaba a la base de datos obteniendo así los valores más actuales (a tiempo real). Se ha mostrado la aplicación, navegando a través de sus diferentes pantallas y se ha expuesto la parte funcional de forma explícita, cerrando así el sistema a tres formado por Arduino - Base de Datos - Android App.

- **Hablar acerca de las futuras propuestas de trabajo que puede plantear el presente proyecto, comentando las posibilidades de mejora que pudiera tener y la aplicabilidad del sistema.**

Aunque a lo largo del proyecto se ha ido dando "pistas" acerca de las "Futuras líneas de trabajo", será en el siguiente capítulo (Capítulo 11) cuando se cumpla este punto en su totalidad.

Una vez hablado acerca de los objetivos, se procede a comprobar el cumplimiento (o no) de la planificación temporal inicial planteada. La actividad completa se dividió en 4 tareas principales. Para la primera, "Sensorización y acondicionamiento", se pensaron unos 25 días teniendo comienzo un lunes, día 2 de mayo. Se ha de recalcar que se dedicaron realmente unos 20, por lo que se redujo el tiempo de realización en 5 días. En la segunda tarea, "Conexión, envío y almacenamiento de datos en la base de datos" se cumplieron con los días establecidos, 21 días. Por el contrario, para la aplicación de Android, se estipularon 25 días iniciales, un valor muy por debajo del valor real. Se utilizaron unos 38 días para poder realizarla, siendo la causa la aparición de una actividad secundaria ajena a este proyecto que impidió al ejecutante (un servidor) poder realizarla en el tiempo establecido inicialmente. Debido a este aumento de días, el tiempo dedicado a "La memoria y el PowerPoint" ha sido menos de lo estipulado, ajustándonos a las fechas de las entregas.

A pesar de no cumplir totalmente con la estipulación temporal inicial, se ha cumplido el objetivo temporal de fin de actividad, llegando a tiempo a la entrega de este proyecto.

Por último, se procede a comentar el grado de satisfacción debido a la realización de este trabajo. Desde un punto de vista objetivo, se hace saber que, exceptuando la parte del sensado, todo lo demás se inició desde conocimiento cero. No se tenía idea inicial acerca de bases de datos, ni de aplicaciones en Android, por lo que comenzar fue difícil. Aún así, se ha disfrutado durante la realización del trabajo, se ha aprendido y, sobretodo, se ha enfrentado ante un problema real que uno se puede encontrar en su futuro trabajo. Ha sido un proyecto muy completo y me siento orgullo de haberlo podido completar. Por tanto se puede decir que el grado de satisfacción es muy alto finalmente.

Capítulo 11. Futuras líneas de trabajo

Finalmente, en este capítulo vamos a exponer lo dicho a lo largo del proyecto acerca de las futuras líneas de trabajo que se podrían tener en cuenta a la hora de mejorar este diseño, introduciéndolas y explicándolas de tal manera que quede claro cuál podría ser el futuro de todo lo realizado hasta el momento.

En primer lugar, se ha de recordar que lo fabricado ha sido solo un prototipo de pruebas. No hemos tenido en cuenta el concepto espacial ya que si se observa con detenimiento la Figura 48, donde se mostraba la estación de sensores al completo, podemos ver que ocupa bastante espacio, pudiendo llegar a abarcar solamente una cuarta parte de toda la placa si hacemos un estudio previo. Por lo que, en primer lugar, se debería hacer un análisis intensivo de espacio para reducir al máximo la superficie del módulo utilizada. También, teniendo en cuenta las necesidades del diseño, se debería considerar la colocación final de los dos elementos principales: Arduino y la placa con los sensores, siendo una posibilidad aconsejable la colocación del Arduino debajo de la placa, reduciendo así el espacio en la horizontal, tal y como se puede ver en la Figura 80, aunque, de nuevo, es solo una posibilidad. Además de todo esto, se debería mirar también un encapsulado funcional de todo el sistema para impedir que se dañe por el agua procedente de la lluvia, esto es, un encapsulado que dejará solamente la parte útil del sensor a la vista y, el resto (patillas, etc.) estuviera impermeable ante situaciones de peligro (agua, etc.).

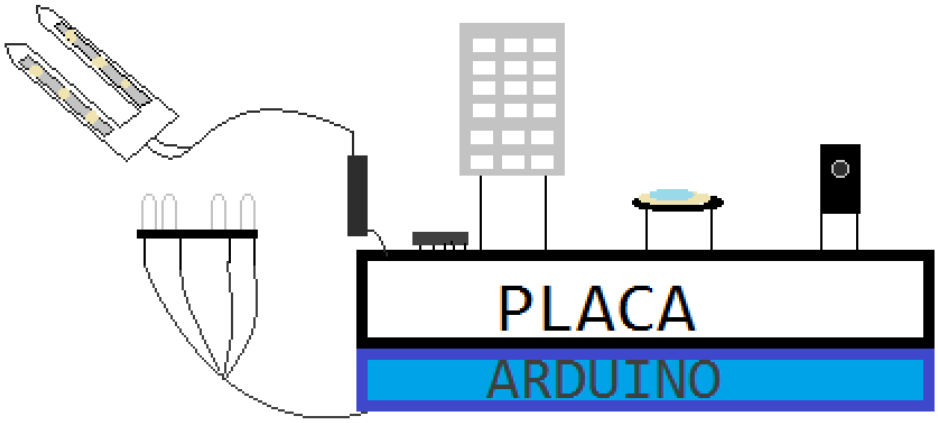


Figura 80. Reducción de espacio en placa y montaje con Arduino

Respecto a la estrategia de comunicación, en este caso hemos utilizado un modelo basado en comunicación por cable para el envío de datos, es decir, mediante la conexión Ethernet del módulo ENC28j60 se ha conseguido enviar todos los datos a la base de datos. En el caso de la distribución óptima para el ejemplo de las 28 tomateras, si la conexión fuera por cable, se necesitarían 2 módulos ENC28j60, cuánto más grande sea el cultivo, mayor será el número de módulos Ethernet. Es por ello que se propone una mejora con respecto a este tema.

Modernizando la comunicación, se podría utilizar un módulo Bluetooth por Arduino, que como bien se ha dicho en el presupuesto, costaban sobre 5€, algo más barato que el módulo Ethernet, además de la ventaja de librarte de los cables de conexión. Así mismo, en vez de un ordenador controlando en base, se podría utilizar una Raspberry Pi ya que, mediante la temporización de envíos de datos de las diferentes estaciones sensoras y activando el envío de cada una en un

tiempo diferente, se podrían manejar todas esas medidas desde un único módulo central, es decir, varias estaciones sensoras podrían conectarse mediante Bluetooth (en tiempos diferentes, no colapsando la comunicación) a la Raspberry Pi, a través de un Arduino intermedio, siendo ésta la encargada de subir los valores medidos a la base de datos. Esto nos permite no tener una comunicación basada en cable, el cuál para la seguridad del sistema debería ir entubado y bajo tierra, evitándonos así un trabajo extra innecesario si se utiliza comunicación Wireless.

Por último, cabe destacar que el fin principal de este proyecto solo era el comienzo de uno nuevo aún más extenso. Cualquier proyecto que haya sido desarrollado con un mínimo de entusiasmo genera nuevas inquietudes en el desarrollador, provocando así la aparición de nuevos campos de ejecución y mejora del mismo. Es por ello que, inicialmente, este proyecto se pensó como el comienzo de una larga carrera hasta llegar a la meta; se podría considerar una primera fase de un proyecto más extenso que no quedará parado aquí. Una vez implementado todo lo dicho en este proyecto, se podría pasar a una segunda fase aún más funcional y también necesaria a la hora de no poder estar presencialmente durante el crecimiento del cultivo: la automatización de las tareas de regado, es decir, la implementación de la parte domótica del sistema.

Para implementar el sistema domótico, en primer lugar, se tendría que instalar una electroválvula para el control del flujo del agua, con dos posiciones (abierto y cerrado), en la boca del tanque de agua usado para esta actividad. Una vez instalada la electroválvula, se tendría que seguir el esquema de la Figura 81 para poder implementar el sistema domótico mencionado:

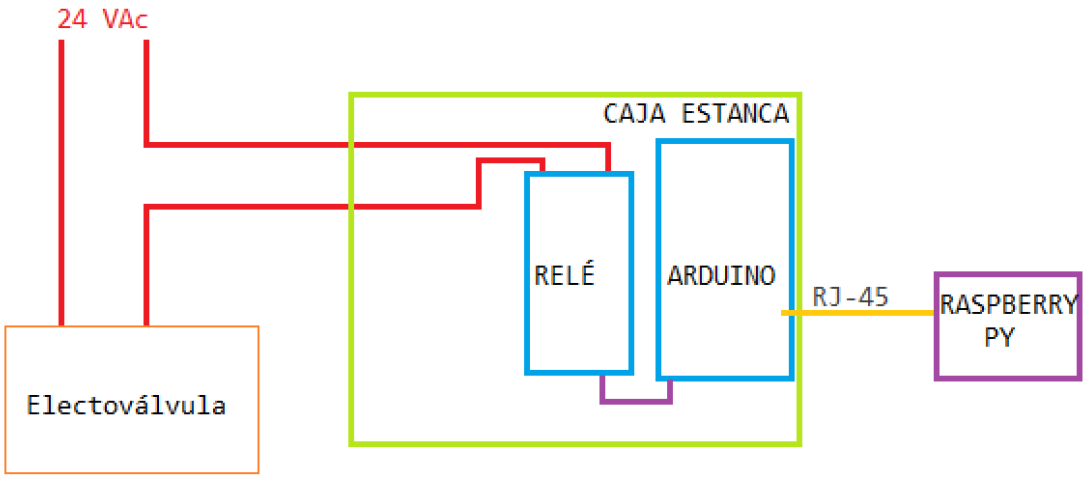


Figura 81. Esquema para la implementación del sistema domótico

Como se puede comprobar en la figura anterior, necesitamos unas Raspberry Pi que será la ejecutora de la orden de regar a partir de recibir la señal de necesidad de regado. Ésta, mediante conexión Ethernet, se conectará con el Arduino intermediario, que servirá de mediador para la activación del relé. Por si no se conoce, el funcionamiento del relé consiste en abrir o cerrar un circuito, por lo que cuando el Arduino recibe la señal "regar" de la Raspberry Pi, éste obliga al relé a que cierre el circuito y, en consecuencia, se abra la electroválvula y se produzca el regado. Además, se podría temporizar el tiempo de riego para tenerlo activado el tiempo que necesitemos o consideremos. Por lo que, en resumen, los Arduinos "medidores" (los que se encuentren en las plantas tomateras) se programarán para enviar en diferentes momentos temporales (cada vez uno) todos los valores necesarios a un Arduino intermediario mediante el uso de módulos bluetooth para, mediante con conexión Ethernet, enviar esos datos finalmente a la Raspberry Pi.

Por otro lado, al estar en fase de pruebas (aplicación beta) no se ha realizado una tarea que se deberá realizar en un futuro si se quiere utilizar plenamente la aplicación. Como se puede comprobar, la app Android se conecta al localhost. Esto es debido a que ambos están trabajando con la misma red (intranet), entonces no tiene problemas a la hora de conectarse al localhost

propio del ordenador. Sin embargo, si nos conectamos fuera de la misma intranet, el firewall del router obviamente no te dejará pasar. Es por ello que, en un futuro con el sistema más avanzado, se tendrá que configurar el router de trabajo (el usado por el ordenador que recibe los datos de Arduino) y pasarle al sistema de la aplicación la IP privada para que ésta pueda funcionar en cualquier lugar.

Con esto se finalizaría este completo proyecto el cual buscaba facilitar la vida al pequeño agricultor que, por la edad u otras causas, no puede visitar diariamente los cultivos que con tanto anhelo ha visto crecer durante años. Además, al tener una aplicación para un Smartphone disponible para realizar el monitoreo y, en un futuro, el control domótico sobre el cultivo, se incentiva el cuidado del mismo (aunque sea a distancia) por parte de los descendientes del agricultor principal en el supuesto de que hiciese falta.

Capítulo 12. Bibliografía

[1] Agricultura Blogspot, "¿Cómo nació la agricultura en el mundo?" http://agricultura03.blogspot.com.es/p/como-nacio-la-agricultura-en-el-mundo.html. [Online].

[2] Juan Manuel Lozano Romero, "Guía para cultivar tomate" http://es.slideshare.net/JuanManuelLozano/guia-para-cultivar-tomate. [Online].

[3] Astromía, "El clima: temperatura, humedad, presión" http://www.astromia.com/tierraluna/elemclima.htm. [Online].

[4] Astrofísica y Física, "¿Qué es el Espectro Electromagnético?" http://www.astrofisicayfisica.com/2012/06/que-es-el-espectro-electromagnetico.html. [Online].

[5] La Huertina de Toni, "Cultivo del tomate: plagas y remedios" http://www.lahuertinadetoni.es/cultivo-del-tomate-plagas-y-remedios/ [Online].

[6] iDESWEB, "MySQL y acceso a una base de datos" http://idesweb.es/proyecto/proyecto-prac10-php-mysql-acceso-base-de-datos [Online].

[7] Código 21, "Descripción de los bloques integrados de App Inventor 2" http://codigo21.educacion.navarra.es/autoaprendizaje/descripcion-de-los-bloques-integrados-de-app-inventor-2. [Online].

[8] EducaCHIP, "Cómo conectar un relé con Arduino y la red eléctrica" http://www.educachip.com/como-conectar-un-rele-con-arduino/. [Online].

[9] UPV, "Aspectos formales de los TFG/TFM" https://www.upv.es/entidades/ETSIAMN/info/AnexoIII.pdf. [Online].

[10] Goilav, Nicolas.; Loi, Geoffrey. "Arduino: aprender a desarrollar para crear objetos inteligentes" *eni-ediciones*, vol. 1, pp. 26–186, 2016.