UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN EN OPCIÓN AL GRADO DE:

INGENIERO EN ELECTRÓNICA DIGITAL Y TELECOMUNICACIONES

TEMA: Diseño e implementación de un prototipo de sistema de control, supervisión de temperatura y humedad, para cultivos caseros bajo invernadero, utilizando el módulo Arduino, en la ciudad de Cayambe

AUTOR: Juan Carlos Caiza Chimarro

TUTOR/ A: Mg. David Cando Garzón

AÑO: 2016

INFORME FINAL DE RESULTADOS DEL PIC

CARRERA:	Electrónica Digital y Telecomunicaciones
AUTOR:	Juan Carlos Caiza Chimarro
TEMA DEL TT:	Diseño e implementación de un prototipo de sistema de control, supervisión de temperatura y humedad, para cultivos caseros bajo invernadero, utilizando el módulo Arduino, en la ciudad de Cayambe
ARTICULACIÓN CON LA LINEA DE INVESTIGACIÓN INSTITUCIONAL:	Tecnología aplicada a la producción y la sociedad
SUBLINEA DE INVESTIGACIÓN INSTITUCIONAL:	Mejoramiento en procesos de cultivo y producción
ARTICULACIÓN CON EL PROYECTO DE INVESTIGACIÓN INSTITUCIONAL DEL ÁREA:	Desarrollo de sistemas automáticos para mejorar la productividad en invernaderos caseros

RESUMEN

El presente proyecto está basado en el diseño e implementación de un prototipo de sistema de control, supervisión de temperatura y humedad, para cultivos caseros bajo invernadero, utilizando el módulo Arduino, en la ciudad de Cayambe.

Para el desarrollo se realizó un estudio e investigación del Módulo Arduino y sus distintas áreas aplicativas, que permitió diseñar un prototipo adecuado, lo cual facilitara a los operadores de cultivo, tener control de la temperatura y humedad dentro del invernadero.

Posteriormente la elaboración de un prototipo de invernadero con sus distintos componentes como sensores y actuadores, además de una aplicación de escritorio para su supervisión y control, el cual puede ser aplicado a distintos tipos de cultivos y formas de los invernaderos.

DESCRIPTORES:

Control Invernaderos

Aplicaciones Arduino

Diseño

Investigación

ABSTRACT

This project is based on the design and implementation of a prototype system for control, monitoring of temperature and humidity for home greenhouses, using the Arduino module, in the city of Cayambe.

To develop a study and research Module Arduino and its different application areas, which allowed design a suitable prototype was made, which facilitate cultivation operators have control of temperature and humidity inside the greenhouse.

Subsequently developing a prototype greenhouse with various components such as sensors and actuators, as well as a desktop application for monitoring and control, which can be applied to different types of crops and forms of greenhouses.

CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

1.1 Problema Investigado

Los llamados “Cultivos Caseros”, como actividad para la ecología y salud, han experimentado un crecimiento notable en los últimos años dentro de los núcleos de población más grandes. Los alimentos cultivados por este método son considerados generalmente de gran calidad, y son numerosos los restaurantes de renombre que cultivan sus propias materias primas en huertos propios y especializados para ofrecer una experiencia diferente a sus clientes.

Desde hace varios años en el campo de los cultivos hortícolas, hay una tendencia, hacia la producción anticipada o totalmente fuera de estación, ha llevado a la puesta a punto de diversos sistemas protectores como los cultivos en invernadero.

En la ciudad de Cayambe, en cultivos a pequeña o gran escala el método preferido para el control climático externo adverso es, “bajo invernadero”.

En la mayoría de los casos controlan y monitorean sus diferentes invernaderos de forma manual, es decir, cuando hay temperatura alta se tendrá que abrir cortinas para que ingrese aire fresco y baje la temperatura, el riego con fertilizante se lo hace manual, midiendo su desarrollo, tomando en cuenta el ciclo de vida de la planta.

El monitoreo y supervisión de tareas en los invernaderos repartidos en diferentes áreas o puntos de la granja de producción, están caracterizados por la dificultad en la toma de mediciones o por la continua necesidad del monitoreo manual de las plantas.

Los invernaderos caseros generalmente, carecen de un sistema para llevar un control automático del monitoreo y control de las plantas y que permita la administración total del mismo o si existe, es demasiado costoso implementarlo.

El presente Proyecto, está dedicado a desarrollar un sistema de monitorización y control de invernadero apoyado en la popular plataforma de hardware libre Arduino, comunicación Ethernet y Procesing para su monitoreo visual.

Este sistema, monitorea las principales variables implicadas en el crecimiento de los cultivos, como pueden ser la humedad del suelo, humedad ambiental y temperatura, obtenidos mediante sensores colocados en el invernadero.

La información enviada desde los sensores será procesada por la aplicación de escritorio, la cual se encargará de mantener la temperatura y humedad del suelo dentro de valores configurados, a través de los actuadores. Además puede aplicarse a cualquier cultivo.

El usuario podrá seleccionar el modo manual para activar riego, ventilación y calefacción si así lo requiriera en cualquier momento.

1.2 Objetivos

1.2.1 Objetivo general:

Implementar un prototipo de sistema de control, supervisión de temperatura y humedad, para cultivos caseros bajo invernadero, utilizando el módulo Arduino, en la ciudad de Cayambe, para mantener en buenas condiciones climáticas que favorecerán el desarrollo de los cultivos, con lo cual aumentara su productividad.

1.2.2 Objetivos específicos:

- Estudiar los elementos necesarios para diseñar el prototipo de sistema de invernadero con sistema automático de temperatura y humedad a través de la plataforma de Arduino
- Diseñar un prototipo que se ajuste a las características necesarias para el adecuado control de temperatura y humedad en un invernadero
- Implementar un prototipo de sistema de invernadero con control automático de temperatura y humedad, el cual utilizara sensores y actuadores y pueda ser implementado en forma real en cultivos bajo invernadero

2. FUNDAMENTACIÓN TEÓRICA Y METODOLÓGICA DEL PRODUCTO O RESULTADO QUE REPRESENTA

2.1 Introducción

Para el desarrollo del presente trabajo fue necesario investigar los parámetros básicos a controlar en un invernadero los cuales son temperatura y humedad, el funcionamiento de los módulos Arduino para medición y control, investigar el funcionamiento del software Processing, en el cual se podrá interactuar con sensores y actuadores, para una adecuada supervisión - control de un invernadero casero.

2.2 Invernaderos

Es una construcción de madera hierro u otro material, cubierta por cristales, provista por lo general de calefacción que, a veces, esta iluminada artificialmente y en donde se pueden cultivar hortalizas, flores y plantas verdes, en épocas en que la temperatura y la luz del lugar serían insuficientes para su crecimiento y su fructificación.

El cristal o plástico usado para un invernadero trabaja como medio selectivo de la transmisión para diversas frecuencias del espectro visible, y su efecto es atrapar energía dentro del invernadero, que calienta el ambiente interior. También sirve para evitar la pérdida de calor por convección. Esto puede ser demostrado abriendo una ventana pequeña del invernadero, la temperatura cae considerablemente. Este principio es la base del sistema de enfriamiento automático auto ventilación. (Alpi, A., & Tognoni, F. , 1991)

2.3 Factores Ambientales en un invernadero

Las principales consideraciones que se debe tener en cuenta cuando se realiza un cultivo es; establecer las condiciones climáticas óptimas para una producción de alta calidad. Entre los principales factores físicos a controlar es la temperatura, humedad del ambiente y humedad del suelo.

2.3.1 Temperatura

Cada especie vegetal, en cada momento crítico de su ciclo biológico, necesita condiciones ambientales óptimas para su desarrollo normal.

La temperatura, influye en las funciones vitales siguientes: transpiración, respiración, fotosíntesis, germinación, crecimiento, floración, fructificación.

Las temperaturas máximas y mínimas que soportan la mayoría de los vegetales están comprendidas entre 0° C y 70° C Fuera de este límite casi todos los vegetales mueren o quedan en estado de vida latente.

Por debajo de determinadas temperaturas, variables para cada especie, y sin que lleguen a temperaturas de congelación, los vegetales detienen totalmente su desarrollo vegetativo a esta temperatura se denomina temperatura mínima letal.

Cuando las temperaturas son altas se produce la coagulación del protoplasma celular y la muerte celular a esta temperatura se denomina temperatura máxima letal.

Las plantas, para un desarrollo correcto de su actividad vegetativa, necesitan que se den unas diferencias de temperatura entre el día y la noche. (Cermeño, 2005).

En la tabla 1, se establece las temperaturas mínimas y máximas biológicas, óptimas para distintos tipos de cultivos.

Tabla 1. Temperaturas para el desarrollo vegetativo de algunos cultivos

Cultivo	Temperatura mínima Biológica en °C	Temperatura máxima biológica en °C	Temperatura optima en °C
Tomate	10	30	20 a 24
Pimiento	13	40	18 a 25
Pepino	10	35	18 a 25
Melón	13	30	20 a 25
Lechuga	6	30	15 a 20
Frutilla	5	30	15 a 18

Fuente: (Cermeño, 2005)

2.3.2 Humedad relativa.

Para un valor constante de vapor de agua en la atmósfera del invernadero (humedad absoluta), a medida que aumenta la temperatura de la atmosfera del invernadero, disminuye la humedad relativa de la misma, También al disminuir la temperatura aumenta la humedad relativa.

La humedad de la atmosfera interviene, en el crecimiento de los tejidos, en la fecundación de las flores, transpiración y en el desarrollo de enfermedades.

La humedad del ambiente influye bastante en el fenómeno de transpiración; cuanto más húmedo este el ambiente menos posibilidades hay de aumentar la evaporación, a no ser que se aumente la temperatura del ambiente.

Cuando la transpiración es intensa, consecuencia de la falta de humedad en el ambiente, puede haber concentración de sales en la parte donde se realiza la fotosíntesis y quedar disminuida esa función.

Cada especie vegetal requiere en el ambiente una cantidad de humedad distinta, variando de unos cultivos a otros como se demuestra en la tabla 2. (Cermeño, 2005)

Tabla 2. Humedad relativa óptima para algunos tipos de cultivos.

Cultivo	Humedad %
Tomate	50-60
Pimiento	50-60
Pepino	70-90
Melón	60-70
Lechuga	60-80
Frutilla	70-80

Fuente: (Cermeño, 2005)

2.3.3 Humedad del suelo

Esta es de gran importancia debido a que el agua constituye un factor importante en la formación, conservación, fertilidad y productividad del mismo, así como para la germinación, crecimiento y desarrollo de las plantas cultivadas.

Para su medición se emplean métodos manuales como bloques de yeso humedecidos, con la cual se puede medir su resistencia eléctrica. (Infoagro, 2015)

2.4 Control de procesos en ambientes

El concepto de control es extraordinariamente amplio y abarca escenarios de complejidad muy diversa. En su concepción más simple, el control alude al gobierno de un sistema por otro sistema.

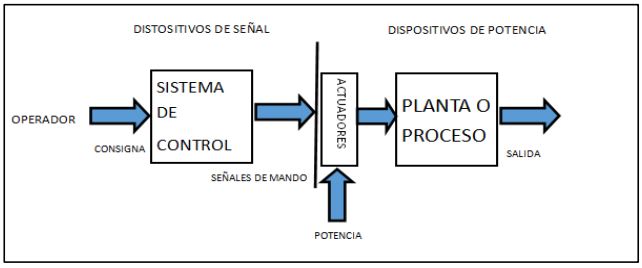


Ilustración 1. Control en lazo abierto

Autor: ((Perez, Alvarez, Campo, Ferrero, & Gustavo, 2004)

El objetivo de un sistema de control es obtener una salida, es decir una respuesta que coincida con las que pretende el operador pero sin que intervenga directamente sobre el sistema. Para ello el operador se sirve de entradas, denominadas variables de control que le permiten especificar la respuesta que necesita el proceso. A partir de estas señales, el sistema de control genera las denominadas señales de mando que son las que actúan sobre la planta con objeto de modificar la salida del proceso. Generalmente, el sistema de control funciona con magnitudes de baja potencia, llamadas señales y gobierna accionamientos o actuadores. Esta idea se detalla en la ilustración 1, donde el conjunto formado por el sistema de control y los accionamientos ejecutan las órdenes dadas por el operador a través de las entradas de consigna. (Perez, Alvarez, Campo, Ferrero, & Gustavo, 2004)

Este tipo de sistemas se denomina en “lazo abierto”, debido a que el sistema de control no recibe ningún tipo de información del comportamiento de la planta, es decir, la salida del proceso no afecta a la acción de control.

En un sistema de control con lazo abierto, a cada consigna corresponde un modo de funcionamiento fijo. Este método se utiliza cuando se conoce de antemano la relación entre la entrada y la salida y no existen perturbaciones de ningún tipo, Una perturbación es una señal que modifica negativamente la salida de un sistema. Si sobre un sistema se producen perturbaciones impredecibles, los sistemas de control en bucle abierto no

pueden corregir la situación ya que el sistema de control no recibe la información sobre este hecho.

Es importante considerar la “perturbación impredecible” ya que si fueran predecibles siempre se podrían compensar dentro del sistema. Cuando se producen perturbaciones impredecibles y se pretende que la planta se comporte de la forma deseada, es necesario informar al sistema de control sobre esta situación.

La ilustración 2, muestra el diagrama de bloques genérico de un sistema de control realimentado. Ahora, el operador fija las variables de consigna o de referencia, establecen el comportamiento deseado, y el control genera las señales de mando para conseguir que la salida del proceso, variable regulada, se mantenga en el valor deseado a pesar de las perturbaciones exteriores. Regular una variable significa mantener esa variable en un valor deseado frente a las influencias externas.

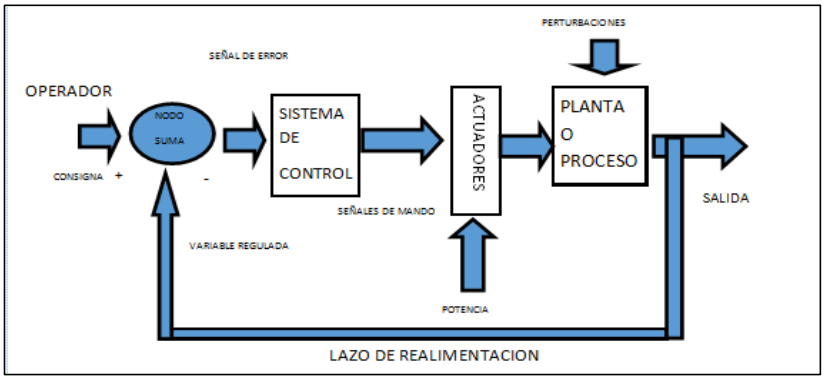


Ilustración 2. Sistema de control realimentado o en lazo cerrado.
Autor:((Perez, Alvarez, Campo, Ferrero, & Gustavo, 2004)

Este tipo de sistemas se denominan “sistemas de control de lazo cerrado” o realimentados, en esta configuración se alimenta al controlador con la denominada señal de error que es la diferencia entre la señal de referencia y la señal de realimentación. La señal de realimentación puede ser la propia señal de salida, variable regulada o una función de esta. Una ventaja de los sistemas realimentados es que desensibiliza el sistema frente a las perturbaciones exteriores o incluso frente a las variaciones internas de los parámetros del sistema.

En el contexto del control de procesos, el interes de la instrumentacion electronica radica en las tecnicas y procedimientos que esta diciplina aporta para medir la variable

de proceso e informar al sistema de control. No se puede controlar una variable de forma mas precisa a como se la puede medir, asi que la medida es un elemento fundamental para el control. En definitiva el conocimiento sobre el estado de un proceso se realiza empiricamente a traves de la medida, y esta labor puede realizarce recurriendo a procedimientos electronicos de instrumentacion diseñando un sistema de medida integrado en el ciclo de control (Perez, Alvarez, Campo, Ferrero, & Gustavo, 2004)

2.4.1 Elementos para el control de procesos

Cuando se habla de instrumentar cualquier sistema fisico se refiere a añadirle todos los sistemas de captacion que sean capaces de leer los parametros fisicos que formen parte de el, la Ilustracion 3, representa los elementos esenciales de un sistema de medida en el contexto del control de procesos.

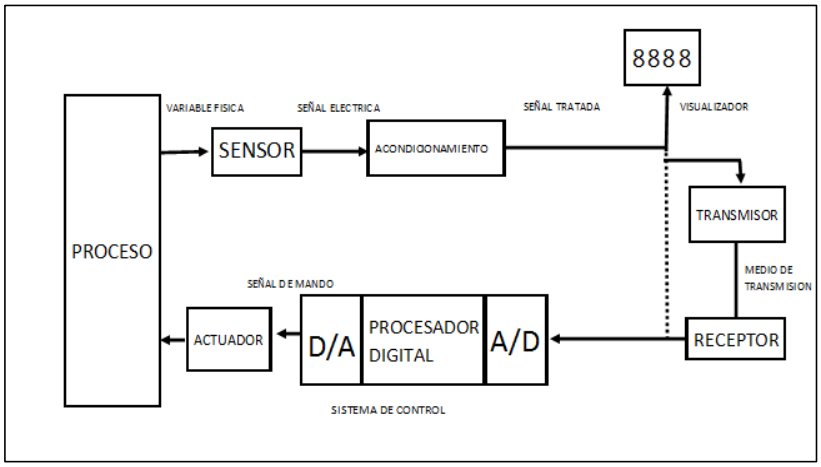


Ilustración 3. Diagrama de bloques de un sistema de medida en contexto de control.
Fuente: (Perez, Alvarez, Campo, Ferrero, & Gustavo, 2004)

Los bloques básicos que se reflejan en la figura son:

- **Sensor**: Captura la variable de proceso, variable física y convertirla en una señal eléctrica.
- **Acondicionamiento de señal**: Trata la señal eléctrica para adaptarla al siguiente bloque del proceso

  El acondicionamiento puede incluirse:

  - Amplificación

- Filtrado
- Conversión de nivel
- Conversión de tensión a corriente y viceversa

- **Visualización:** unidad que presenta la información al operador
- **Sistema de control:** procesa los datos de acuerdo al algoritmo de control y genera la señal de mando.

  Los sistemas de control pueden ser:

  - Analógicos: Emplean circuitos analógicos para realizar el control.
  - Digitales: Basados en procesadores digitales, como los Arduino
- **Actuadores:** conjunto de dispositivos que modifican la respuesta del sistema.
- **Sistemas de trasmisión remota:** permite transmitir la información entre la planta y el sistema de control en el caso de que estén alejados uno del otro. (Perez, Alvarez, Campo, Ferrero, & Gustavo, 2004)

2.5 Control con módulo Arduino

Arduino es una plataforma open-source", cuya placa tiene un microcontrolador, las cuales son capaces de leer y escribir puertos, como activar salidas y o entradas para luces, presión sobre un botón, motor, LED, contactor, relé, etc. Todo ello definido en un conjunto de instrucciones programadas a través de su entorno de desarrollo.

El 'hardware' de las placas consiste normalmente en un microcontrolador AVR y puertos de entrada/salida. Se suelen emplear microcontroladores como el ATmega328, el ATmega2560, o el ATmega1280, entre otros, debido a su sencillez y bajo coste su uso es muy popular. (Arduino.cc, 2016)

2.6 Comunicaciones con Arduino

Arduino tiene una serie de facilidades para la comunicación con un ordenador, Arduinos, o micro controladores. El ATmega 2560 proporciona cuatro UART hardware para TTL (5V) de comunicación serie. Un ATmega16U2 canaliza a uno de ellos sobre el USB y proporciona un puerto COM virtual para software al equipo (máquinas de Windows tendrá un archivo .inf, El software de Arduino incluye un monitor serie que permite enviar datos desde la placa y viceversa. Los LEDs RX y TX de la placa parpadean cuando se están transmitiendo datos a través de ATmega8U2/ATmega16U2 chip y la conexión USB al ordenador. La biblioteca (Software Serial), permite la

comunicación en serie en los pines digitales, también soporta la comunicación SPI. (Arduino.cc, 2016).

2.6.1 El protocolo de Ethernet y el estándar IEEE 802.3

El protocolo Ethernet fue desarrollado a principios de los años setenta por Xerox para conectar estaciones de trabajo. A comienzos de los 80, DEC, Intel y Xerox completaron el estándar Ethernet a 10 Mbps para transmisión LAN a través de cable coaxial, estándar LAN que fue la base del IEEE 802.3. El estándar ha sido mejorado y revisado en pocos años, se ha especificado para cable coaxial, cableado de par trenzado y fibra óptica. Las versiones de alta velocidad se aprobaron en 1995 (Ethernet de alta velocidad a 100 Mbps) y en 1998 (Ethernet Gigabit a 1000 Mbps).

El estándar original se definió para LAN en bus de cable coaxial en que las transmisiones se difunden a través del medio en bus según el protocolo MAC CSMA-CD (acceso múltiple sensible a la portadora con detección de colisiones). Así, una estación con una trama a transmitir espera hasta que el canal este desocupado. Cuando esto ocurre, la estación transmite al tiempo que continua escuchando el medio con objeto de detectar colisiones que podrían producirse si otras estaciones comenzasen a transmitir. Si se produce una colisión, la estación aborta la transmisión y planifica un intervalo de tiempo aleatorio posterior en el que intentara de nuevo la transmisión de la trama. Si, por el contrario, no tiene lugar colisión alguna dentro de un intervalo de tiempo igual a dos veces el tiempo de propagación, la estación tiene constancia de haber tomado con éxito el canal, y sus transmisiones se recibirán en el resto de estaciones, las cuales no intentaran transmisión alguna hasta que finalice la de la estación primera. (Leon & Indra, 2002)

2.6.2 Software de Arduino

La plataforma Arduino tiene un lenguaje basado en C/C++ y por ello soporta las funciones del estándar. Sin embargo, es posible utilizar otros lenguajes de programación y aplicaciones más populares en Arduino como Java, Processing, Python, Matlab, Perl, Visual Basic, Lab View, etc. Esto es posible debido a que Arduino se comunica mediante la transmisión de datos en formato serie que es algo que la mayoría de los lenguajes anteriormente citados soportan.

Para los que no soportan el formato serie de forma nativa, es posible utilizar software que traduzca los mensajes enviados por ambas partes para lograr una comunicación compatible.

Se puede tener la posibilidad de interactuar con Arduino, mediante esta gran variedad de sistemas y lenguajes, puesto que dependiendo de cuales sean las necesidades del problema que va a resolver, puede aprovechar de la gran compatibilidad de comunicación que ofrece. El entorno para el desarrollo de Arduino es sencillo, además puede descargarse gratuitamente desde su página oficial para distintos sistemas operativos. Arduino está implementado en Processing, un lenguaje similar a Java.

Está formado por una serie de menús, una barra de herramientas con botones para las funciones comunes, un editor de texto donde se escribe el código, un área de mensajes y una consola de texto, la ilustración 4, se puede apreciar la interfaz gráfica del software de Arduino. (Herias, FC. Gomez, G.G., Baeza, J:P., Bravo, C.J., 2015)

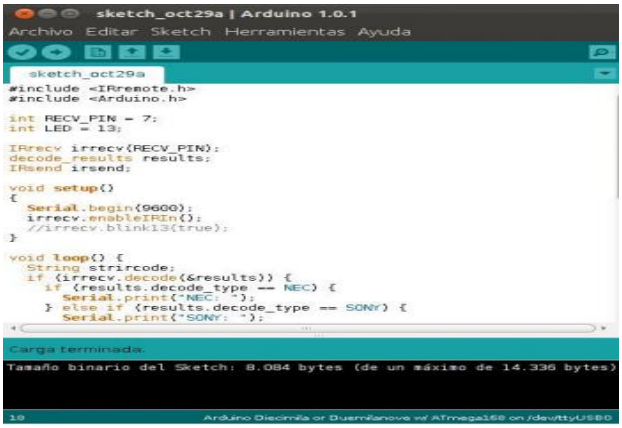


Ilustración 4. Interfaz del Software Arduino
Fuente: (Arduino.cc, 2016)

La parte más importante se encuentra en la pestaña de herramientas. Desde aquí se puede configurar para que pueda comunicarse con la placa Arduino. Aquí en esta barra de herramientas se selecciona el Arduino dependiendo de con cual se esté trabajando.

En el menú principal los botones y sus funciones son los siguientes:

- Verificar: Comprobar y compilar el código.

- Cargar: Compila el código y lo carga en la placa

- Nuevo: Crea nuevo sketch.

- Abrir: Abre un sketch guardado.

- Guardar: almacena en disco cambios realizados en el sketch.

- Monitor Serial: abre una nueva ventana en la que se puede comunicar bidireccionalmente vía serie con la placa, es decir, leer la información que envía o se la puede proporcionar. La ilustración 5 muestra ejemplo de esta ventana.

Ilustración 5. Ventana del monitor serie
Fuente: (Arduino.cc, 2016)

- Editor de texto: En esta área se escribe la implementación (denominada por el programa sketch) para poder cargarla en la placa Arduino. El programa tiene tres partes. La primera consiste en inclusión de librerías y la declaración de constantes o variables globales que se podrán utilizar en cualquier función del programa. La segunda es el método setup, que es encargado de inicializar los dispositivos conectados a la placa y será ejecutado al iniciar el sistema. La tercera parte consiste en el método loop, que ejecutará su código en forma continua, es decir, en modo bucle. Aquí es donde se escribirá la lógica de la placa Arduino, como el lenguaje es muy similar a C es posible crear otros métodos para separar bloques funcionales y dejar ordenado el programa.

- Área de mensajes: Muestra la situación al haber utilizado uno de los botones comunes en el programa.
- Consola de texto: Aparecerán con mayor detalle los eventos del área de mensajes. (Herias, FC. Gomez, G.G., Baeza, J:P., Bravo, C.J., 2015)

2.7 Interfaz de salida con processing

Processing, es un lenguaje de programación y entorno de desarrollo integrado de código abierto el cual está basado en Java, que sirve como medio para la enseñanza y producción de proyectos multimedia e interactivos de diseño digital. Fue iniciado por Ben Fry y Casey.

Processing es desarrollado por artistas y diseñadores como una herramienta visual alternativa al software propietario. Puede ser utilizado tanto para aplicaciones locales así como aplicaciones para la web. Se distribuye bajo la licencia libre o GNU

Se puede ver que la interface se parece a la de Arduino, Ilustración 6.

Ilustración 6. Pantalla para programación del control visual Processing
Fuente: Processing.org

2.7.1 Dispositivos compatibles con Arduino

Para conseguir las características de un sistema de control es necesario que además del órgano central que controle el sistema se tenga a disposición sensores que puedan recoger datos sobre el ambiente del invernadero. Dependiendo de estos datos el sistema debe ser capaz de comunicarse con los actuadores para controlar el ambiente del invernadero.

También deben existir elementos con los que el usuario pueda comunicarse con el sistema y pueda hacer los cambios oportunos manualmente. Los dispositivos estarán conectados mediante cables o directamente acoplados a la placa Arduino. Algunos de ellos disponen de librerías que debe adjuntar al programa para poder usar las utilidades que contengan. Para ello se añade la carpeta de la librería en la carpeta libraries del entorno de desarrollo de Arduino. Al principio del código del sketch se incluye la librería con la línea: #include

Para utilizar los sensores y actuadores digitales se debe tener en cuenta que solo tiene dos posibles valores, HIGH representa el nivel alto y LOW el nivel bajo. En el caso de los analógicos su uso es levemente más complejo pero también más configurable ya que tiene que leerse/escribir un voltaje de 0 a 5 voltios que se representa en 10 bits (lectura) o en 8 bits (escritura), es decir la tensión puede tener 1024 (lectura) o 256 (escritura) valores. (Herrador, 2009)

2.7.2 Sensores

Un sensor es un dispositivo o elemento capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables generalmente eléctricas. Las variables de instrumentación pueden ser: temperatura, distancia, aceleración, inclinación, desplazamiento, presión, fuerza, torsión, humedad, movimiento, etc. Una magnitud eléctrica puede ser una resistencia eléctrica (como en un detector de temperatura resistivo), una capacidad eléctrica (como en un sensor de humedad), una tensión eléctrica (como en un termopar), una corriente eléctrica (como en un fototransistor) (Areny, 2004)

2.7.3 Actuadores

Son dispositivos capaces de transformar energía eléctrica, neumática o hidráulica en la activación de un proceso, con la finalidad de generar un efecto sobre lo que se está automatizando. Este recibe la orden del o controlador y en función a ello, genera la orden para activar un elemento final de control como, por ejemplo, una válvula, ventilador, motor, calefactor, etc.

Existen varios tipos de actuadores como son:

- Electrónicos
- Hidráulicos
- Neumáticos

- Eléctricos

2.8 Descripción del proceso investigado

Las actividades de monitoreo y supervisión se las realiza manualmente, en la mayoría de invernaderos caseros en la ciudad de Cayambe, en la actualidad no se cuenta con un sistema autónomo para dichas actividades a bajo costo.

2.9 Propósito de los Objetivos

Para el desarrollo de este sistema como objetivo principal se planteó, construir un prototipo de invernadero casero el cual de forma autónoma tendrá controlados los factores ambientales de temperatura y humedad además de monitorearlos a través de una aplicación de escritorio en un computador.

Los objetivos específicos se crearon para implementar un cultivo determinado y adecuar las condiciones ambientales durante todo el ciclo del desarrollo de la planta.

2.10 Hipótesis

Cuando se implemente el sistema prototipo de control y supervisión de temperatura y humedad casero, utilizando módulos Arduino, se demostrará que se puede desarrollar un sistema automático a bajo costo y muy flexible, se puede adecuar para cualquier cultivo determinado en el Norte de la Provincia de Pichincha.

Variable Independiente: Implementación de un sistema prototipo automático de control y supervisión de temperatura y humedad en un invernadero casero

Variable dependiente: Se puede construir un sistema automático para supervisión y control de invernaderos caseros de bajo costo y flexibles que se acomoden a cualquier tipo de cultivo.

2.11 Fundamentación Teórica

Para el desarrollo de este prototipo se utilizó como guía la tecnología de placas pre fabricadas Arduino, pues tomando como ejemplo se guía en el modelo, Además de placa Ethernet conectada a un router el cual hará el enlace entre el computador y los sensores y actuadores.

2.12 Metodología de Investigación

Para el desarrollo de este sistema se realizaron varios métodos de investigación que permitieron avanzar con cada etapa propuesta dentro del mismo.

2.12.1 Método de Análisis

Este método se utilizó para realizar la etapa inicial, recopilando la información necesaria de todos los elementos electrónicos posibles a utilizarse, para aprovechar de mejor manera sus características.

2.12.2 Método de Modelación

Este método se utilizó para realizar el diseño tanto del Hardware como de Software, esta es una parte primordial en todo el proceso pues de aquí parte todo el sistema.

2.12.3 Método Experimental

Se utilizó el método experimental ya que por la naturaleza del proyecto se debió realizar pruebas y corrección de errores.

En la parte investigativa se utilizó la técnica de investigación de la entrevista, realizada a varias familias de distintos sectores de la zona norte de la Provincia de Pichincha cantón Cayambe, con el fin de conocer su opinión acerca de este sistema. El formato que se utilizó para realizar las entrevistas se presenta a continuación; en el anexo 1, se puede observar las entrevistas realizadas.

2.13 Formato de resultados de encuesta

A continuación se muestran los resultados de las entrevistas realizadas resumidas en la tabla 3, realizadas a diez propietarios de invernaderos caseros en la zona norte de la provincia de Pichincha.

Pregunta 1. ¿Dispone de un invernadero que controle el ambiente interior en forma automática?

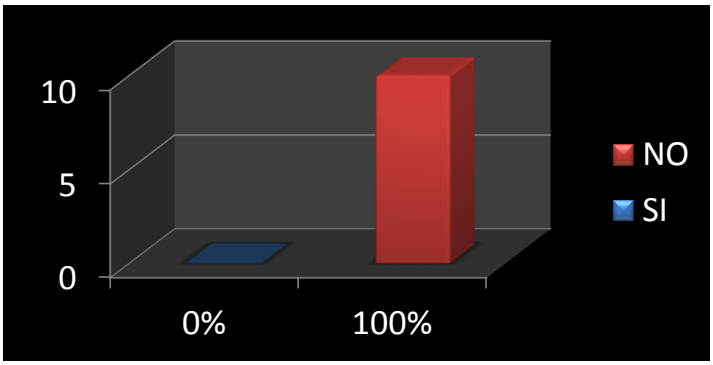


Ilustración 7. Disponibilidad de Invernadero Automático

Fuente: El autor

Interpretación: De la encuesta realizada, 10 personas responden que no disponen de algún dispositivo que controle en forma automática el ambiente de sus invernaderos casero, es igual al 100 %, Ilustración 7

Pregunta 2. Piensa que la tecnología puede facilitar sus actividades diarias en su invernadero, de alguna manera.

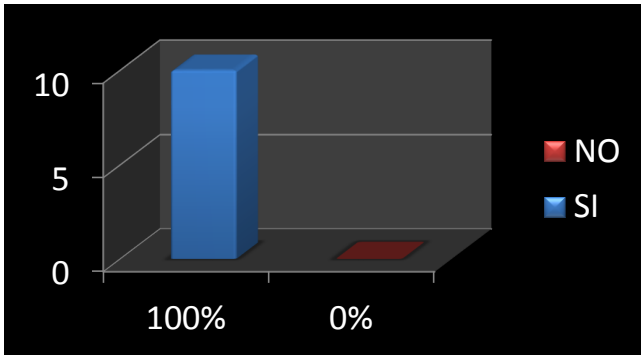


Ilustración 8. Encuesta si facilitaría las actividades

Fuente: El autor

Interpretación: De la encuesta realizada, 10 personas, el 100% de los encuestados, piensa que la tecnología puede facilitar sus actividades diarias en su invernadero, de alguna manera, Ilustración 8.

Pregunta 3. ¿Cómo controla la temperatura y humedad de su invernadero?

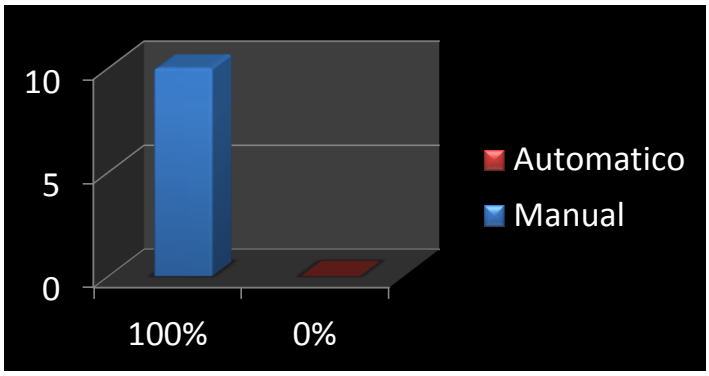


Ilustración 9. Encuesta como controla su temperatura

Fuente: El autor

Interpretación: De la encuesta realizada, 10 personas, el 100% de los encuestados, controla la temperatura y humedad en forma manual, Ilustración 9.

Pregunta 4. ¿Le gustaría tener en su invernadero un sistema que permita controlar y supervisar la temperatura y humedad de su invernadero de manera automática y remota?

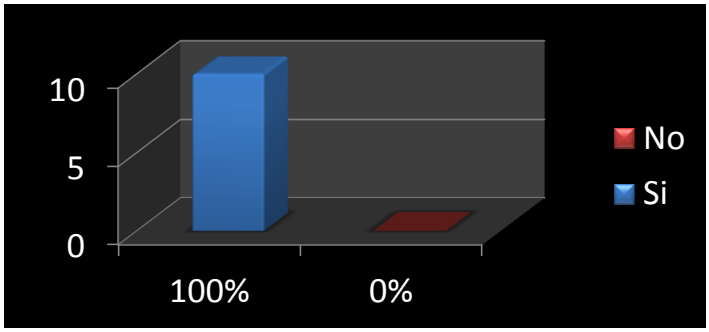


Ilustración 10. Encuesta si desearía tener un invernadero automático

Fuente: El autor

Interpretación: De la encuesta realizada, 10 personas, el 100% de los encuestados, indica que le gustaría tener un sistema automático de control de temperatura y humedad, Ilustración 10.

Pregunta 5. ¿Le parece que este sistema le ahorrara tiempo y controlara el proceso de crecimiento de las plantas de manera óptima?

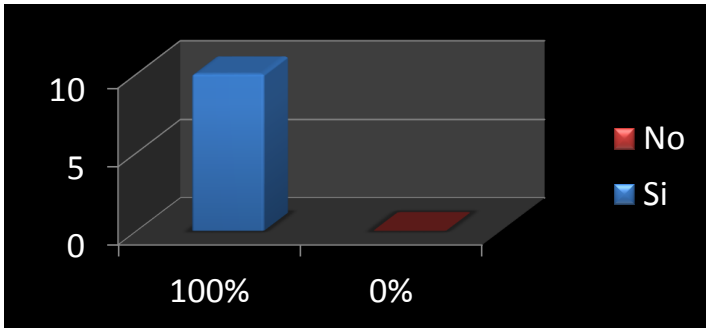


Ilustración 11. Encuesta si cree que optimizara actividades

Fuente: El autor

Interpretación: De la encuesta realizada, 10 personas, el 100% de los encuestados, cree que optimizaría actividades y ahorraría tiempo, Ilustración 11.

Pregunta 6. ¿Invertiría en la implementación del control y supervisión de temperatura y humedad en su invernadero?

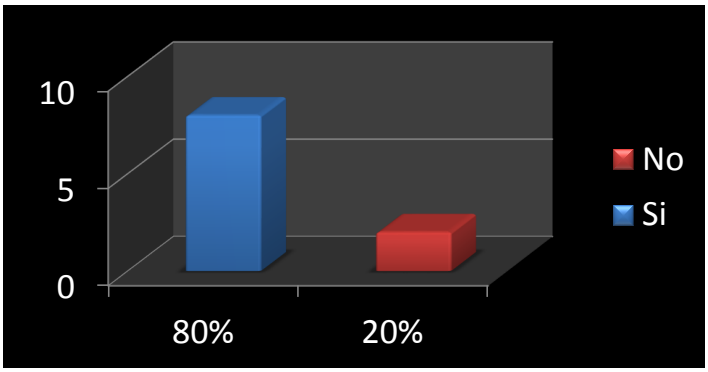


Ilustración 12. Invertiría en el dispositivo automático

Fuente: El autor

Interpretación: De la encuesta realizada, 8 personas, el 80% de los encuestados, si invertiría en la automatización de su invernadero, mientras que 2 personas es decir el 20% de los encuestados no invertirían, ilustración 12.

2.14 Análisis Integral

Se realiza el análisis en la Tabla 3, la cual contiene las preguntas y respuestas realizadas a propietarios que tienen invernaderos caseros.

Tabla 3. Estadístico tabla de análisis integral

PREGUNTA	RESPUESTA	NUMERO	PORCENTAJE
1. ¿Dispone de un invernadero que controle el ambiente interior en forma automática?	SI	0	0%
	NO	10	100%
2. Piensa que la tecnología puede facilitar sus actividades diarias en su invernadero, de alguna manera.	SI	10	100%
	NO	0	0%
3. ¿Cómo controla la temperatura y humedad de su invernadero?	MANUAL	10	100%
	AUTONOMO	0	0%
4. ¿Le gustaría tener en su invernadero un sistema que permita controlar y supervisar la temperatura y humedad de su invernadero de manera automática y remota?	SI	10	100%
	NO	0	0%
5. ¿Le parece que este sistema le ahorrara tiempo y controlara el proceso de crecimiento de las plantas de manera óptima?	SI	10	100%
	NO	0	0%
6. ¿Invertiría en la implementación del control y supervisión de temperatura y humedad en su invernadero?	SI	8	80%
	NO	2	20%

Fuente: El autor

2.15 Resultados que se esperan del Proyecto

Con la construcción del sistema prototipo de supervisión, control de temperatura y humedad se espera demostrar que si es posible realizar un sistema automático de bajo costo, además los propietarios si están dispuestos a invertir en el dispositivo por cuanto ayudarían a estabilizar el clima y la humedad en parámetros adecuados para el normal desarrollo de los cultivos.

3. PRESENTACIÓN DE RESULTADOS

3.1 Propuesta de Solución al Problema

El propósito de este proyecto es el de construir un prototipo de control y supervisión de temperatura y humedad para cultivos bajo invernaderos caseros con ajuste de temperaturas de pendiendo a qué tipo de cultivo se requiera.

Además se implementa un monitoreo en tiempo real para controlar esos parámetros en forma manual o automática, desde una aplicación de escritorio de un computador mediante el modo grafico en Processing, con comunicación Ethernet.

3.2 Diseño General del Sistema Prototipo Automático de control

Para la elaboración del sistema prototipo se utilizaron cinco etapas que son: fuente de poder, circuitos de entrada, control electrónico con Arduino, comunicación Ethernet, Aplicación de escritorio en lenguaje Processing. Ilustración 13.

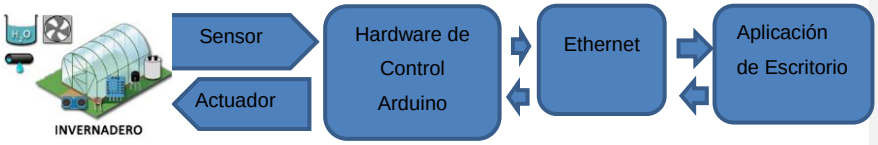


Ilustración 13. Diagrama de control del sistema

Fuente: El autor

3.2.1 Etapa de Fuente de Poder

Esta etapa es la encargada de proveer la energía necesaria para el funcionamiento del sistema, para la alimentación del Arduino se utilizó un adaptador reajustable de 7 a 12 voltios

Para el suministro de energía a actuadores se utiliza el modulo, SMPS600 SW Long Power Supply, con las siguientes características:

El Diseño de referencia CC CA SMPS funciona con un rango de tensión de entrada universal de 110VAC o 220 VAC seleccionables y produce tres tensiones de salida (12 V, 3,3 V y 5 V). La capacidad nominal de salida continua de la diseño de referencia es de 300 vatios. Este diseño de referencia se basa en una estructura modular.

En la Ilustración 14, se muestra la tarjeta con sus distintos componentes: 1. EMI Filter 2. Single-Phase Converter 3. Multi-Phase Converter 4. Primary Side Controller 5. Secondary Side Controller 6. Bridge Rectifier 2 3 4 5 6 7. PFC Boost Converter 8. ZVT Full-Bridge Converter 9. Synchronous Rectifier 10. 12V Output (Intermediate Bus) 11. 3.3V Output 12. 5V Output

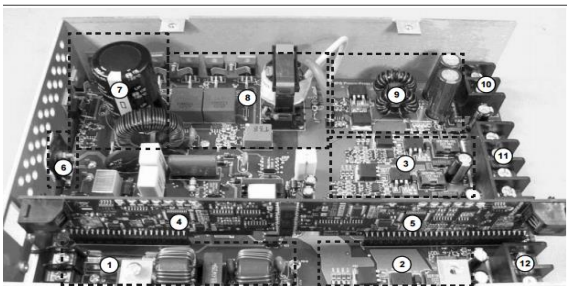


Ilustración 14. Tarjeta de alimentación SMPS600 Power Supply
Fuente: (Microchip, 2016)

3.2.2 Circuitos de Entrada

En esta etapa se considera como circuitos de entrada, a todo elemento o conjuntos de elementos que hacen posible la introducción de alguna variable al sistema, como por ejemplo los sensores utilizados, de temperatura, humedad relativa y humedad del suelo, los sensores utilizados entregan una señal análoga la cual será procesada por el micro controlador del Arduino,

Los sensores siempre que estén activados estarán tomando continuamente la situación actual del proceso, y es el servidor o la placa Arduino quien lee esta información y decidirá cómo actuar.

Las entradas al micro controlador pueden ser digitales o analógicas:

Los digitales tienen que ser inicializados como pin de salida con el método pinMode(numeroDePin, OUTPUT), para poder obtener una lectura de los datos, se usa el método digitalRead(numeroDePin).

Los analógicos no requieren esta fase de inicio y para leer con analogRead(número de pin). Es recomendable asignar a una variable la lectura recibida por los métodos para evitar tener que llamar a la misma función en caso de necesitarse de nuevo, debido a

su bajo coste y compatibilidad con el módulo Arduino se utiliza el DTH11, para medir temperatura y Humedad Relativa y el HL69 para medir humedad el suelo

3.2.2.1 Módulo DTH11, sensor de temperatura y humedad relativa

El sensor de temperatura y humedad DHT11, cuenta con una salida de señal digital calibrada. Mediante el uso de la exclusiva señal de adquisición digital de temperatura y detección de humedad, asegura una alta fiabilidad y excelente estabilidad a largo plazo. Este sensor incluye una medición de la humedad de tipo resistivo y un componente de medición de temperatura NTC, Ilustración 15,

Los 8-bit del microcontrolador, ofrece una excelente calidad, respuesta rápida, anti-interferencias, capacidad y la rentabilidad.

Cada elemento está estrictamente calibrado en el laboratorio. Los coeficientes de calibración se almacenan en programas de la memoria OTP, que son utilizados por el proceso de detección de la señal interna del sensor.

La interfaz en serie de un solo cable hace que la integración del sistema sea rápida y fácil. Su pequeño tamaño, tiene un bajo consumo de energía, el componente es de 4 pines de paquete pasador de una sola fila.

Este módulo es capaz de representar digitalmente la humedad ambiental medida en porcentaje, además de la temperatura en grados centígrados.Tiene una precisión decimal y dispone de su propia librería que contiene los métodos para recoger sus mediciones. (Arduino.cc, 2016)

Ilustración 15. Sensor de temperatura y Humedad DTH11
Fuente: (Hitek, 2015)

3.2.2.2 Módulo HL69 para medición de humedad del suelo

Una solución a bajo coste es el módulo HL-69, un sensor de humedad de suelo, para determinar el nivel de humedad del suelo utiliza la conductividad entre sus dos terminales, Ilustración 16.

Ilustración 16. Modulo HL-69 Sensor de Humedad del Suelo
Fuente: (Hitek, 2015)

El Módulo HL-69, sensor de humedad de suelo, el cual consiste en dos placas separadas entre sí por una distancia determinada. Ambas placas están recubiertas por una capa de material conductor. Si hubiera humedad en el suelo se creará un puente, lo que será detectado por el circuito de control con un amplificador operacional, que será el encargado de transformar la conductividad que registra a un valor analógico el cual podrá ser leído por Arduino. Ilustración 17. (Hitek, 2015)

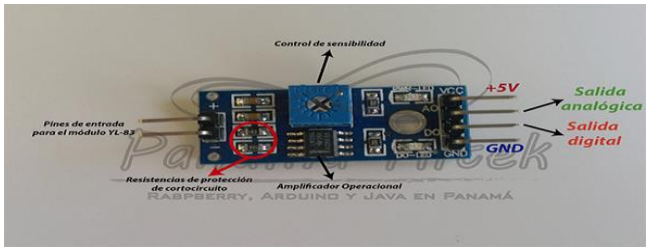


Ilustración 17. Modulo acondicionador de señales análogas
Fuente: (Hitek, 2015)

La salida digital entregará un pulso bajo cuando haya conductividad suficiente entre las puntas. El umbral de disparo se puede calibrar al mover el potenciómetro del circuito de control.

En la salida analógica el nivel de voltaje dependerá directamente de cuanta humedad tenga en el suelo. Dependiendo de cuanta conductividad (agua en el suelo) haya entre las puntas del módulo, variará el valor entregado por Arduino (entre 0 y 1023).

Para medir la humedad del suelo se utiliza el módulo HL-69, utiliza la conductividad entre sus dos terminales para determinar el porcentaje de humedad en el suelo.

3.3 Determinación del dispositivo de control a utilizarse

Para esto se ha elaborado la Tabla 4, en la que se evalúa los elementos de control que pudieran servir para la ejecución del mismo, Existen diferentes placas de Arduino, por lo que la primera tarea del trabajo es elegir la adecuada que llegue adaptarse al proyecto.

A continuación se detallan los principales tipos de Arduino y sus características principales, tabla 4.

Tabla 4. Tabla comparativa entre distintos tipos de Arduino

Características de Arduino	UNO	Mega 2560	Leonardo	DUE
Tipo de microcontrolador	Atmega 328	Atmega 2560	Atmega 32U4	AT91SAM3X8E
Velocidad de reloj	16 MHz	16 MHz	16 MHz	84 Mhz
Pines digitales de E/S	14	54	20	54
Entradas analógicas	6	16	12	12
Salidas Analógicas o PWM	0	14	7	2 (DAC)
Memoria de programa (Flash)	32 Kb	256 Kb	32 Kb	512 Kb
Memoria de datos (SRAM)	2 Kb	8 Kb	25 Kb	96 Kb
Memoria auxiliar (EEPROM)	1 Kb	4 Kb	1 Kb	0 Kb

Fuente: (Arduino.cc, 2016)

Por tener mayor número de entradas salidas, mayor capacidad de memoria de programa y mayor velocidad en manejar los procesos, se elige utilizar Arduino Mega 2560 para el control electrónico del prototipo.

3.4 Control Electrónico con Arduino Mega 2560

Arduino Mega 2560 es una placa electrónica basada en el microcontrolador Atmega2560. Tiene 54 entradas / salidas digitales, 14 de las cuales se pueden utilizar como salidas PWM, 16 entradas analógicas, 4 UARTS (puertos serie de hardware), un cristal de 16 MHz oscilador, una conexión USB, un conector de alimentación, una cabecera ICSP, y un botón de reinicio, Ilustración 18.

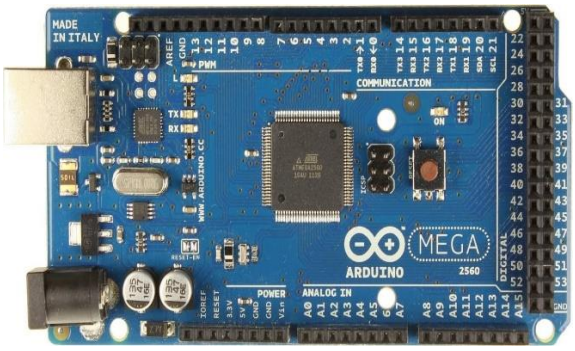


Ilustración 18. Arduino Mega 2560

Fuente: (Arduino.cc, 2016)

Contiene todo lo necesario para activar el microcontrolador; basta con conectarlo a un ordenador con un cable USB o la corriente con un adaptador de CA a CC o una batería. La Mega es compatible con la mayoría de los shield para el Arduino Duemilanove o Diecimila.

La dirección (entrada o salida) de los pines digitales así como la decisión de cuales son configurados como PWM se efectúa en el código 'sketch' del usuario.

Cada uno de los 54 pines digitales se puede emplear como entrada o salida usando las funciones digitalWrite, digitalRead y pinMode de la librería Arduino. Los pines operan a 5 V y cada uno de ellos recibe o provee 20 mA, pudiendo llegar a 40 mA como máximo.

Varios de estos pines, además, pueden tener funciones específicas como las siguientes:

- Interrupciones externas: Se pueden configurar los pines 2, 3, 21, 20, 19 y 18 (de mayor a menor prioridad) para activar una interrupción si se encuentran a nivel bajo, si hay un flanco de subida, si hay un flanco de bajada o si lo hay de cualquiera de los dos. Otras placas Arduino cuentan con un par más de opciones.
- PWM: Los pines 2-13 y 44-46 pueden generar una señal PWM empleando la función *analogWrite*. Las señales PWM son señales binarias que conmutan con un determinado periodo de tiempo entre ambos niveles. La frecuencia de esta señal es de 490 Hz. El pin sigue generando la misma señal hasta que se llama otra vez a *analogWrite*, *digitalRead* o *digitalWrite* sobre el mismo pin.

- La placa cuenta con 16 entradas analógicas, cada una con una resolución de 10 bits (0-1023).
- Por defecto, el valor analógico de entrada se mide desde 0 V hasta 5 V, pero se puede alterar empleando la función *analogReference* y el pin AREF, dependiendo si se quiere usar una referencia interna o externa. (Arduino.cc, 2016)

El controlador Atmega que usa Arduino, lleva incluido un conversor análogo digital (A/D) de 6 canales. Tiene una resolución de 10 bits. El uso principal de estos pines por los usuarios de Arduino es para la lectura de sensores analógicos, estos pines tienen también la funcionalidad de los pines de entrada-salida de propósito general (GPIO) (al igual que los pines 0 al 13). Consecuentemente, si un usuario necesita más pines de propósito general de entrada y salida, y no se está usando ningún pin analógico, estos pines pueden usarse como GPIO.

La mayoría de los usuarios usarán las resistencias pullup cuando se use un pin analogo como digital. La función utilizada es analogRead():Lee el valor de tensión en el pin analógico que se ha programado. La placa Arduino posee 6 canales (8 canales en el Mini y Nano y 16 en el Mega) conectados al conversor analógico digital de 10 bits. Esto significa que convertirá voltajes entre 0 y 5 voltios a un número entero entre 0 y 1023.Esto proporciona la resolución en la lectura de: 5 voltios / 1024 unidades, es decir, 0.0049 voltios (4.9 mV) por unidad. El rango de entrada puede ser cambiado usando la función analogReference(). El conversor tarda aproximadamente 100 microsegundos (0.0001 segundos) en leer una entrada análoga, por lo que se puede llevar una tasa de lectura máxima aproximada de 10000 lecturas por segundo. (Arduino.cc, 2016)

3.5 Comunicaciones

Con la comunicación bidireccional entre el servidor y el cliente Arduino, se requiere que este logre transmitir las órdenes y recibir las actualizaciones del sistema controlado por el Arduino Mega 2560. Por lo tanto se requirió de un método para conectar el Arduino que garantizara la transmisión de los datos, por lo que se decidió implementar la comunicación por medio de módulo Ethernet ENC28J60 y el Router TP-LINK, Ilustración 19.

Es necesaria una infraestructura de red que soporte la comunicación entre los diferentes subsistemas que componen este proyecto, en este caso empleando una red Ethernet de área local para el correcto funcionamiento del sistema.

Ilustración 19. Comunicación entre Arduino, módulo ethernet y router

Fuente: El autor

La comunicación entre el dispositivo Arduino y la aplicación de escritorio utiliza el protocolo UDP sobre Ethernet a través la biblioteca UDP *Hypermedia.net* para *Processing* y *EthernetUDP.h* para Arduino. Se ha elegido este protocolo de red por su buen ajuste a las características técnicas de este proyecto, siendo las principales expuestas a continuación:

- Intercambio rápido de información usando un solo paquete UDP, lo que evita el establecimiento de una conexión previa (*handshaking*) y la sobrecarga asociada al uso del protocolo TCP (20 bytes para el header en TCP, 8 bytes usados en el header UDP), con el correspondiente ahorro en el tráfico de datos enviados por la red.
- La pérdida de un paquete en la red no supone un problema crítico, puesto que el envío de datos y órdenes se realiza de forma casi continua. En vez de solicitar el reenvío si se emplease el protocolo TCP, los datos recibidos en el servidor son reemplazados por la siguiente lectura con valores prácticamente idénticos a los del paquete perdido. En el caso del envío de órdenes se presenta una situación similar, ya que son enviadas al hardware de forma continuada.

3.5.1 Comunicación Ethernet a través de ENC28J60

El modulo, ENC28J60 que proporciona conectividad Ethernet a cualquier micro controlador, como es el caso de Arduino Atmega. En el mercado se encuentran diferentes presentaciones de este módulo, algunas vienen en formato de Shield, con una tarjeta más pequeña pero se cablea fácilmente al Arduino, Ilustración 20.

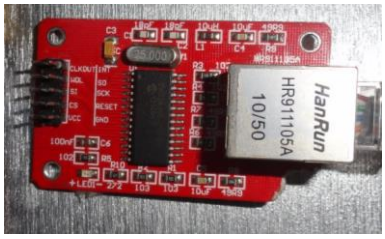


Ilustración 20. Modulo Ethernet ENC28j60
Fuente: El autor

Cabe mencionar que la comunicación entre el módulo Ethernet y el Arduino se lleva acabo utilizando un protocolo que se llama SPI o Serial Peripheral, Interface SPI es un protocolo que se utiliza para comunicar un micro controlador con otro en forma serial síncrono y con periféricos a distancias cortas. Para hacer una conexión SPI siempre habrá un dispositivo maestro (usualmente un micro controlador) que controlará uno o varios periféricos (esclavos), se utilizan por lo generar 3 líneas de conexión y una de selección que son:

- MISO (Master In Slave Out): Utiliza el esclavo para enviar datos al maestro.
- SI o MOSI (Master Out Slave In): Envía datos del maestro al esclavo.
- SCK (Serial clock): Utiliza pulsos de reloj para sincronizar la comunicación
- CS o Select.: Es usada por el master para habilitar o deshabilitar determinado periférico.

Los micros controladores Atmel, tienen las 4 líneas para usar protocolo SPI las cuales obviamente están presentes en los pines. (Hitek, 2015)

3.6 Etapa de salidas

Las salidas están determinadas mediante un módulo de 8 relés apto acoplados, los cuales activaran los distintos actuadores indistintamente del voltaje requerido, Ilustración 21.

Los actuadores controlaran en el caso de desvíos en el rango de trabajo. El voltaje necesario para el funcionamiento de actuadores es: foco de calefacción 110VAC, ventilador 12 VDC, electro válvula 110 VAC.

a) b) c) d)

Ilustración 21. Actuadores: a) Ventilador. b) Cortina. c) Foco. d) electroválvula

Fuente: El autor

Para el control de motor de la cortina, se decide utilizar la tarjeta L298, Ilustración 22, que es un puente H dual diseñado para conducir cargas inductivas tales como relés, solenoides DC y motores paso a paso. Permite conducir de dos Motores de corriente continua, el control de la velocidad y dirección de cada uno de ellos de forma independiente, las características son las siguientes:

- Voltaje operativo de 4V a 35V
- L298N controlador Motor, Conductor 2 DC
- Max 2A de corriente por canal o 4A Max
- Parada de funcionamiento libre y la función de freno
- Chip: ST L298N
- Fuente de alimentación lógica: 5v
- Potencia máxima: 25w

Ilustración 22. Tarjeta L298 Dual H, para control de motor de cortina

Fuente: STMicroelectronic

3.6.1 Esquema completo de la placa

La ilustración 23, muestra el diseño completo de la placa realizada en el programa Proteus.

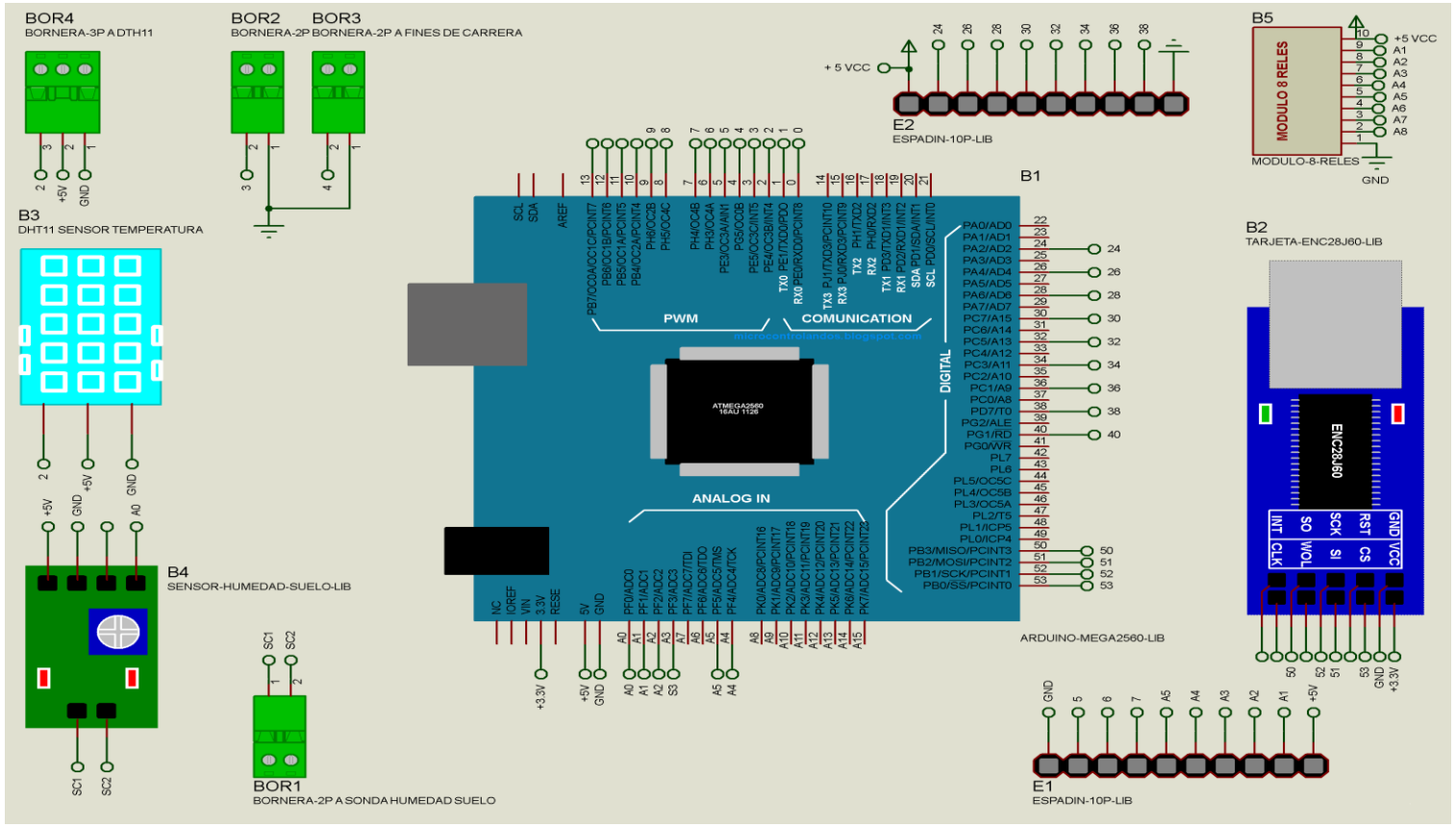


Ilustración 23. Esquema de circuito de control automático, de temperatura y Humedad

Fuente: El autor

3.6.2 Circuito impreso del equipo

El circuito impreso se realizó en Proteus - Ares, el cual cuenta con los pines de conexión del equipo para su implementación en la baquelita, la ilustración 24, muestra las pistas de conexión de la placa principal del sistema.

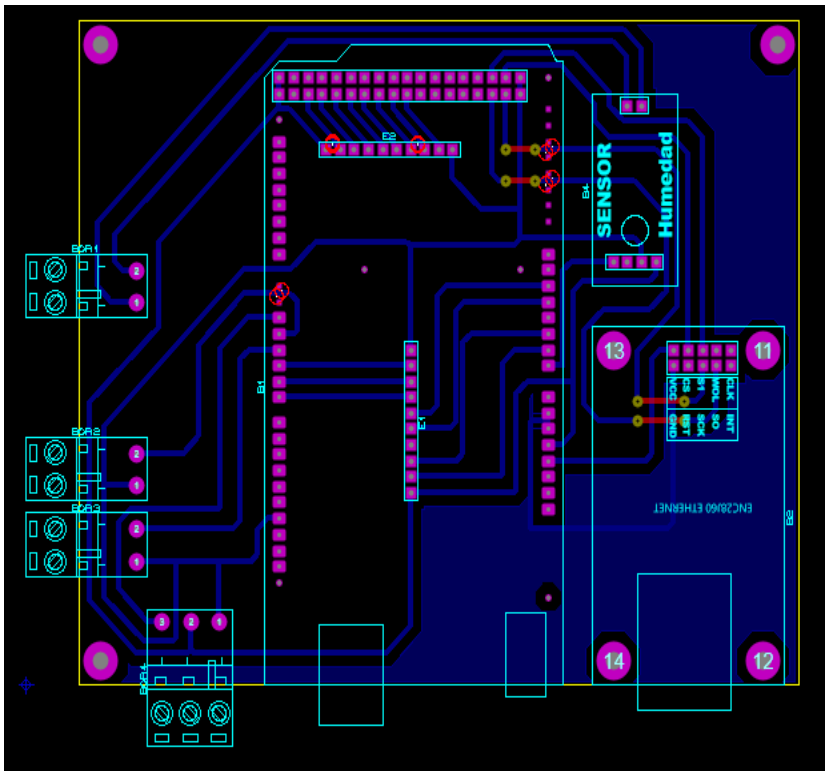


Ilustración 24. Placa en Baquelita del circuito de control

Fuente: Proteus – El autor

3.7 Implementación del Equipo

Para elaborar la placa principal se realizaron las pruebas necesarias en el programa Proteus- Ares, para luego poder implantar este proceso en una baquelita, se realiza la implementación con sus placas shield respectivas, Ilustraciones: (25, 26, 27).

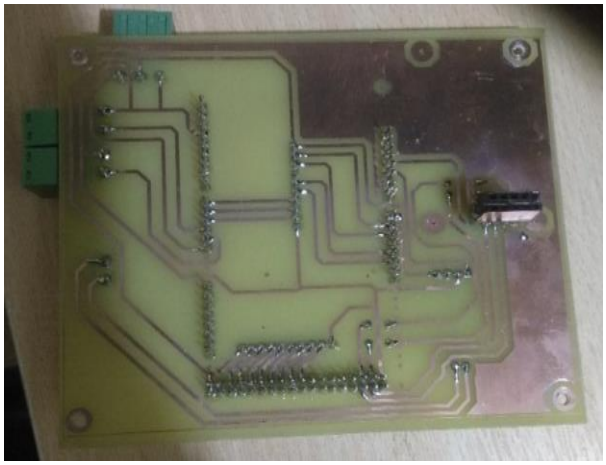


Ilustración 25. Placa Impresa Vista frontal

Fuente: El autor

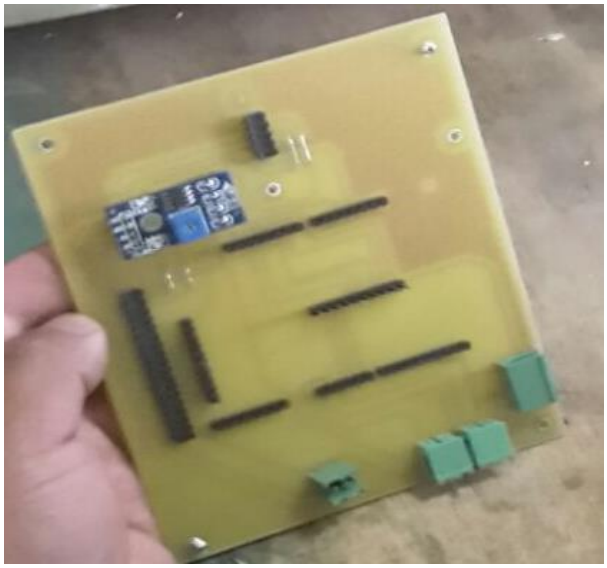


Ilustración 26. Circuito impreso vista posterior

Fuente: El autor

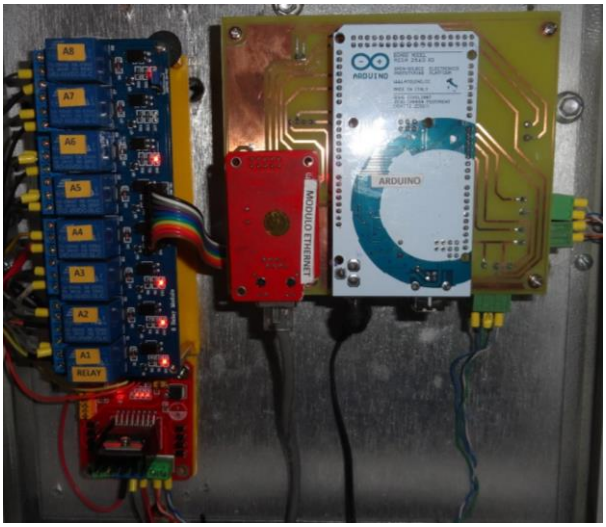


Ilustración 27. Circuito impreso implementado real vista frontal

Fuente: El autor

3.8 Software del sistema

3.8.1 Programación en Arduino y processing para aplicación de Escritorio

El desarrollo Arduino, está constituido por un editor de texto para escribir el código, un área de mensajes, una consola de texto, una barra de herramientas con botones para las funciones comunes, y una serie de menús. Permite la conexión con el hardware de Arduino para cargar los programas y comunicarse con ellos, el programa implementado se detalla en el Anexo 3.

El lenguaje de programación Processing, entorno de desarrollo integrado de código abierto basado en Java, de fácil utilización, proporciona un ambiente grafico amigable para la comprensión de los distintos procesos, se debe seguir una secuencia para el enlace con el Arduino, como inicializar parámetros botones gráficos, comunicación, conexión SQL, ejecutar secuencia de programa principal, en el Anexo 3, se detalla el programa implementado en este lenguaje.

La aplicación de escritorio se podría considerar como el corazón del sistema, Ilustración 28, puesto que se encarga de las funciones principales, como son la comunicación directa con el hardware Arduino y la interacción con el usuario. Desde esta aplicación se controlan los rangos aceptables de las variables monitorizadas (humedad ambiental, humedad del

suelo, temperatura interior, temperatura exterior y nivel de agua para riego) así como el modo de funcionamiento de los sistemas de riego y ventilación (ON, OFF, AUTO).

Estas condiciones y el estado actual de los sistemas de riego y ventilación son almacenados en ficheros de texto dentro del servidor del programa de descarga gratuita, XAMPP, cada vez que se modifican los parámetros en la aplicación para mantener la consistencia de configuraciones y estados entre los subsistemas *front-end*. La interfaz se ha tratado de simplificar todo lo posible para hacerla intuitiva y sencilla a la vez que funcional.

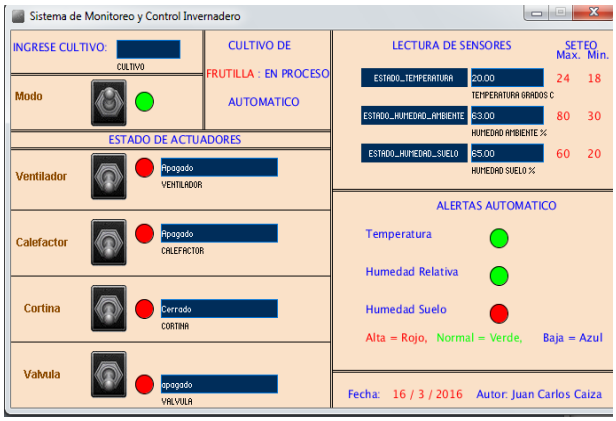


Ilustración 28. Visualización de salidas en aplicación de escritorio

Fuente: El autor

3.8.2 Crear la base de datos en SQL con XAMPP

Primero se debe descargar XAMPP “xampp-win32-1.7.4-VC6-installer”, desde internet, al instalar el ejecutable solo se debe poner siguiente y seleccionar todos los campos de instalación luego entrar desde un navegador c33omo el google chrome o cualquier otro, la dirección que se pone simplemente es http://localhost , presionar cargar, la cual deberá mostrar la página de acceso al administrador del servidor xampp, previamente hay que en el panel de administración hacer correr al servidor apache para poder hacer el proceso de manejo en la web, listo y completado, Ilustración 29.

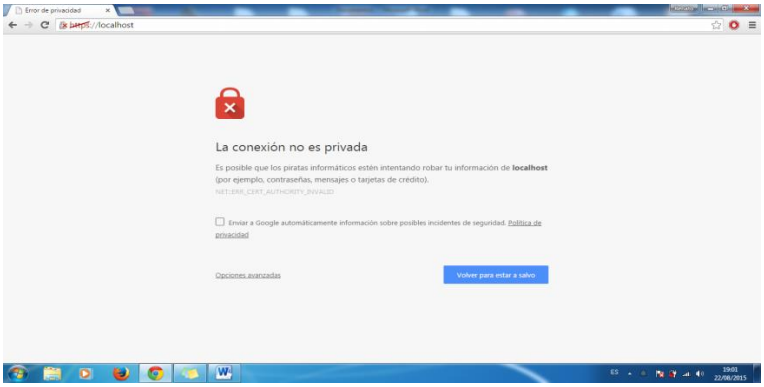


Ilustración 29. Programa XAMPP para crear base de datos

Fuente: XAMPP

Luego se dirigirá, a las opciones avanzadas para poder evadir los permisos de conexión porque no se ejecuta como una conexión segura de internet, Ilustración 30.

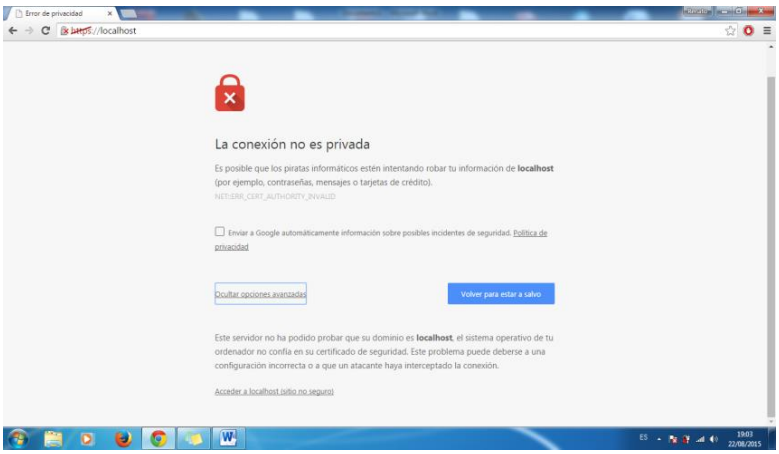


Ilustración 30. Permisos para ejecución XAMPP

Fuente: XAMPP

A continuación accederá, a la pantalla principal del sistema de administración de xampp por la página web, Ilustración 31.

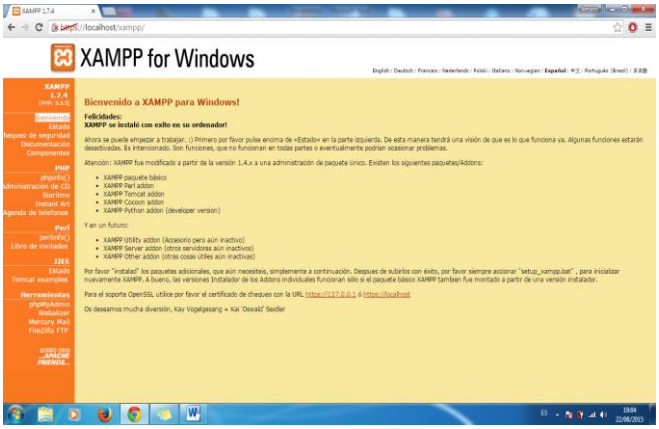


Ilustración 31. Pantalla principal XAMPP

Fuente: XAMPP

Se dirigirá a la barra de herramientas y presionara phpAdmin, luego ingresar a la sección en donde podrá crear la base de datos de forma gráfica, Ilustración 32.

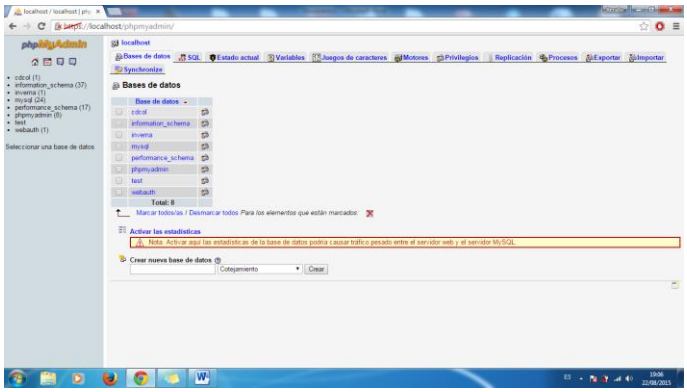


Ilustración 32. Pantalla grafica de herramientas de bases de datos

Fuente:XAMPP

En esta sección se creara una nueva base de datos con el nombre correcto en este caso ya está generada y se llama “Inverna”, Ilustración 33.

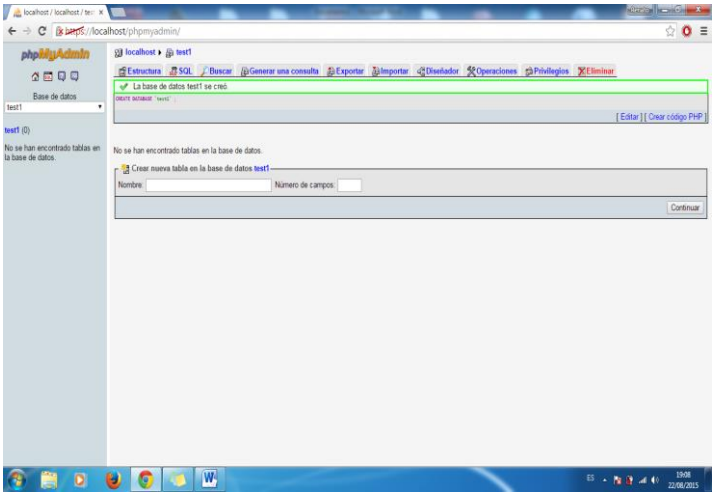


Ilustración 33. Pantalla para creación de base de datos

Fuente: XAMPP

Por ejemplo, se crea una base de datos que se llama test1, luego al presionar en la opción crear una nueva tabla y se escribe una llamada por ejemplo “Contenedor” y especificar el número de campos para el manejo de los cultivos o lo que se vaya a realizar, a continuación se presenta la configuración en modo web de como configurar los datos de los registros y por defecto poner sus nombres y una clave primaria en el primer campo que por diseño de base de datos se debe llamar “Id” o identificador de cada registro de datos, Ilustración 34.

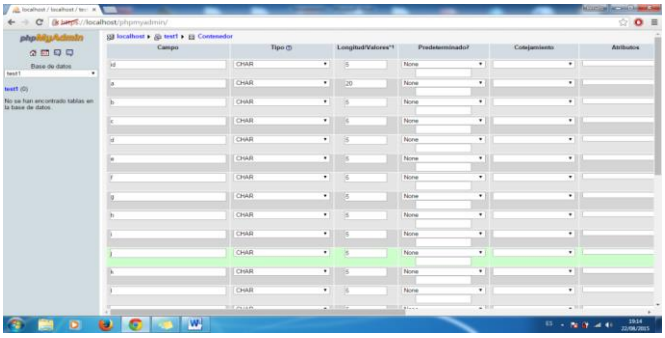


Ilustración 34. Pantalla de configurar de datos

Fuente: El autor

Se deberá llenar los campos dependiendo del tipo de dato se va a obtener de la tabla y sus respectivos nombres al que corresponde cada celda, se definen todos los campos de tipo como “char”, por ser de fácil manejo y conversión al momento de administrar la base de datos.

A continuación se presionará el botón Grabar para completar este proceso y tener lista la base de datos simple y empezar a ingresar los datos desde el mismo administrador Xampp o desde comandos dependiendo como se realice estos accesos, en este caso se debe integrar a Processing una librería que permita realizar estas conexiones de bases de datos, ingresar al Processing y descargar la respectiva librería, Ilustración 35.

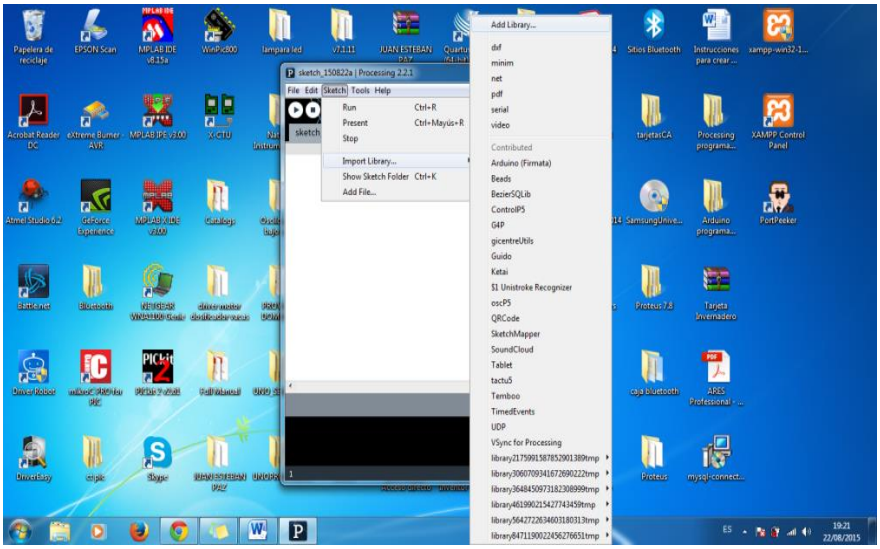


Ilustración 35. Pantalla de carga hacia Processing

Fuente: Processing.org

Seleccionar la casilla e introducir “SQL”, luego instalar la “BezierSQLib”, para conectar a la base de datos creada en el servidor Xampp, Ilustración 36, En el anexo 2 se explicará los comandos básicos para manejar la base de datos.

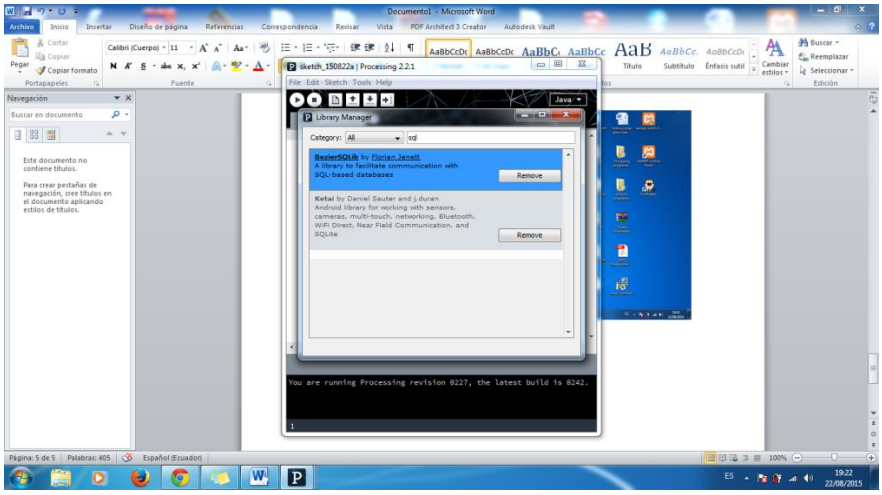


Ilustración 36. Pantalla para conexión a base de datos

Fuente: Processing.org

3.9 Validación del sistema

Se realiza las pruebas en forma manual, observando que todos los actuadores se enciendan correctamente.

Luego se hacen las pruebas en Modo Automático, se ingresara cultivo y verificara el funcionamiento real del dispositivo, para ello se ubicaron todos los componentes de acuerdo al diseño establecido en la placa de componentes, además se estableció la comunicación entre los sensores y el monitor del computador.

3.10 Evaluación y pruebas

De acuerdo a las pruebas realizadas, se obtiene que la estructura del dispositivo debe ser la adecuada y equilibrada, para que no genere un mal funcionamiento y esto altere el correcto desarrollo del dispositivo, a continuación se indica que tipos de pruebas se realizó durante todo el proceso de desarrollo del proyecto.

Para realizar la evaluación y validación del funcionamiento del sistema implementado se han realizado las siguientes pruebas:

Verificación del funcionamiento del Sistema de Seguridad implementado en el invernadero.

Verificación del funcionamiento del Sistema del lado de los sensores y actuadores, se verifica la programación de parámetros en la base de datos Inverna, los cuales son:

- Para el cultivo de……(programable)……
- Duracion del ciclo…..…… (programable)…
- Temperatura maxima……(programable)
- Temperatura minima ……..(programable)
- Humedad relativa maxima ……(programable)
- Humedad relativa minima …….(programable)
- Humedad del suelo baja…….(programable)
- Humedad del suelo alta……(programable)

La programacion de actuadores se realiza de la siguiente manera:

- Si temperatura dentro del invernadero es mayor que la temperatura programada en la base de datos para determinado cultivo, debera activar salidas para encender ventilador y abrir cortina, la cortina tiene un interruptor de carrera el cual determinara el fin de la apertura, activa tambien una alerta luminosa en el tablero
- Si temperatura dentro del invernadero es menor que la temperatura programada en la base de datos para determinado cultivo, verifica que este cerrada cortina y encendera la calefaccion.
- Si humedad relativa es mayor o menor respecto a lo programado en la base de datos, daran una alerta visual
- Si humedad del suelo es mayor a la humedad programada en la base de datos, se visualizara una alerta luminosa en el tablero y la interfaz grafica
- Si humedad del suelo es menor a la humedad requerida para determinado cultivo, enciende alerta luminosa y activa la valvula de riego.

3.11 Pruebas finales

Para realizar las pruebas, se elige un cultivo de la base de datos, como es frutilla, con los siguientes parámetros a controlar:

3.11.1 Parámetros de cultivo de prueba en automático

El cultivo de frutilla tiene los siguientes parametros los cuales estan alamcenados dentro de la base de datos, Ilustracion 37.

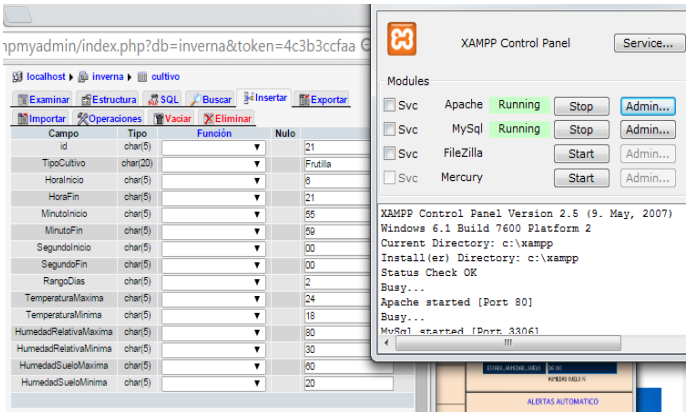


Ilustración 37. Base de datos cultivos de Prueba.

Fuente. El autor

Se selecciona el cultivo en la pantalla de interface programada en Processing y luego poner en automático, se observa que la temperatura y humedad está dentro del rango de trabajo programado, por lo tanto no enciende ningún actuador, Ilustración 38.

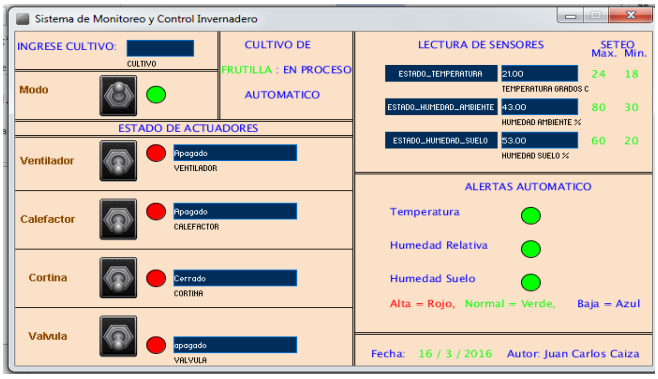


Ilustración 38. Ejecutando el programa automático, selección frutilla

Fuente: El autor

3.11.2 Pruebas de Temperatura

Se realizan pruebas en alta y baja temperatura, con respecto a un rango especifico de temperatura en determinado cultivo determinado por la base de datos, en este caso el cultivo de frutilla.

- Alta temperatura

Al calentar intencionalmente el sensor se visualiza que se enciende el ventilador y se abre la cortina, Ilustración 39.

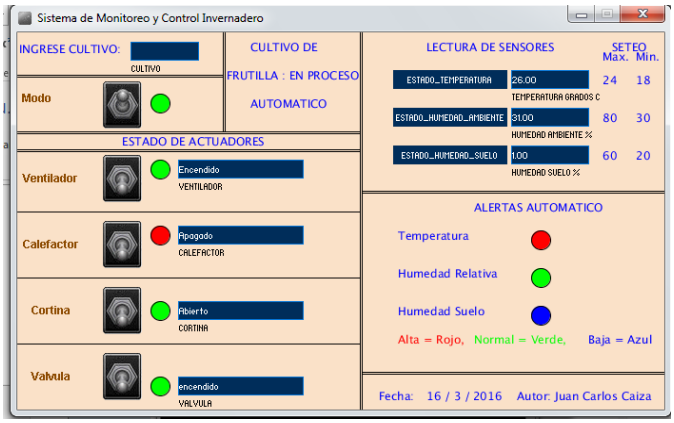


Ilustración 39. Prueba de alta temperatura

Fuente: El autor.

- Baja Temperatura

Al enfriar intencionalmente observar que se cierra la cortina y se enciende el calentador, Ilustración 40.

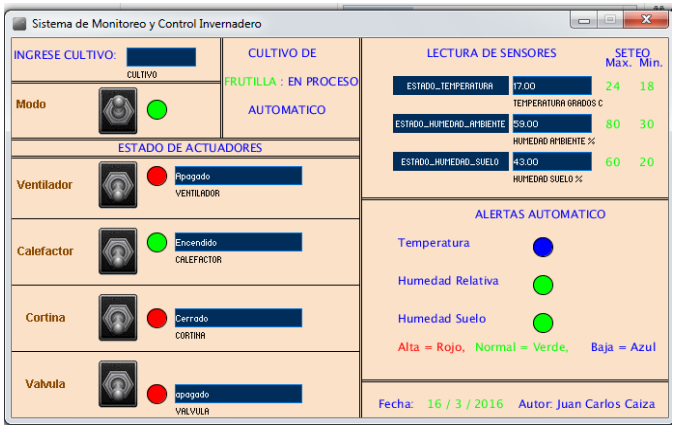


Ilustración 40. Prueba en baja temperatura

Fuente: El autor

3.11.3 Prueba de Humedad del suelo

Se realizan pruebas de humedad el suelo, se toman datos de la base de datos del cultivo para establecer el rango de trabajo, fuera de este rango de trabajo se tendrá alta o baja humedad

- Baja Humedad

Se realizan las pruebas respectivas en ambiente seco, se encenderá la válvula de agua automáticamente y en exceso de humedad dará una alerta, Ilustración 41.

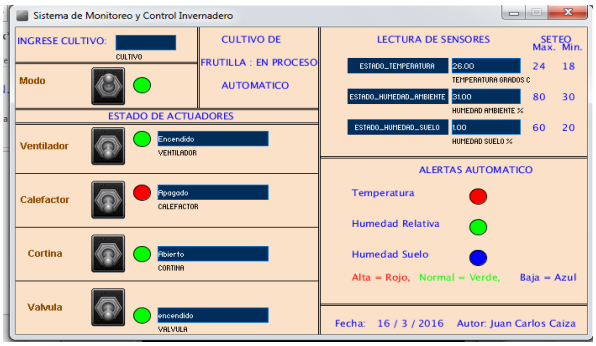


Ilustración 41. Prueba de baja humedad

Fuente: El autor

- Prueba de alta Humedad del suelo

Simplemente alertara el sistema, Ilustración 42.

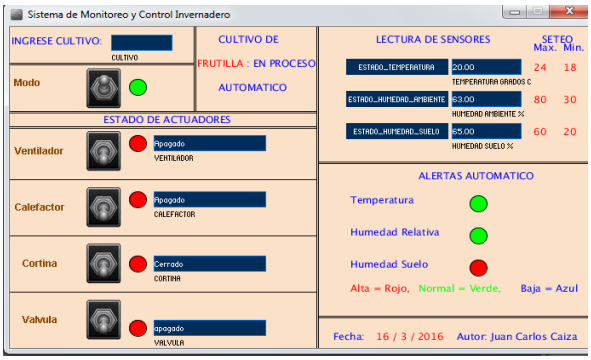


Ilustración 42. Prueba de Humedad Alta

Fuente: El autor

3.12 Evaluación Técnica

Se lleva a cabo el siguiente CHECK LIST del funcionamiento del dispositivo, detallado en la tabla 5

Tabla 5. Evaluación Técnica

Procedimiento	**Funcionamiento**		
	Correcto	**Incorrecto**	**Observaciones**
Encendido de placa	X		Se observa encendido de placa Arduino
Se visualiza parámetros en computador	X		Se visualiza en Proccesing ejecutando el programa
Se visualiza estado de actuadores	X		Se visualiza en Proccesing ejecutando el programa ejecutando actuadores de acuerdo a programación establecida
Análisis de funcionamiento habitual en condiciones de manejo diarias.	X		Se realiza pruebas por varias horas, se ejecutan de acuerdo a lo establecido

Fuente: El autor

3.12.1 Análisis de resultados de las pruebas finales:

Se realiza las pruebas y verificación del desempeño del dispositivo, hay que tomar en cuenta que todas estas pruebas han sido necesarias para realizar un balance de funcionamiento del dispositivo en condiciones normales de uso diario sin ninguna novedad o anomalía, esto garantiza un correcto trabajo del sistema.

Además se comprueban que los actuadores para compensar temperatura; funcionan de acuerdo a lo requerido.

Se enciende calefacción al detectarse dentro del invernadero baja temperatura. Se activa ventilador y apertura de cortina al detectarse altas temperaturas,

También se confirma que se activa electroválvula para controlar la humedad baja del suelo y alerta en alta humedad del suelo, humedad relativa del aire solo dará alerta.

3.13 Factibilidad económica y costo de implementación

El siguiente proyecto de investigación y desarrollo pretende demostrar la viabilidad y rentabilidad de invertir en la implementación de un prototipo de invernadero, contratación del personal necesario para llevar a cabo las operaciones e instalaciones necesarias.

En el mercado existen varios tipos de controladores para invernaderos, pero el sistema Arduino es el más económico. A continuación en la Tabla 6, se detalla el costo de la implementación.

Tabla 6. Costo de materiales y mano de obra.

Equipos	Precio Unitario (USD)	Precio Total (USD)
Placa Arduino Mega 2560	38	38
Modulo Ethernet ENC28j60	20	20
Módulo de 8 Relay	15	15
L298 Dual Bridge	13	13
Sensor temperatura	12	12
Sensor de humedad	12	12
Motoreductor	9	9
Ventilador 12 V	10	10
Foco calefactor	8	8
Sensor de fin de carrera	5	5
Fuente de alimentación	15	15
Placa, Borneras, otros	60	60
Mano de obra 2 personas 40 horas efectivas medio SMV	250	250
Total		**467 USD**

Fuente: El autor

3.14 Análisis de retorno de inversión

Al comparar las actividades que se realizan en forma manual y automático, para controlar los parámetros ambientales más importantes en un invernadero. Se determina que para el método manual deberá invertir en mano de obra sin tomar en cuenta la efectividad del cuidado ya que puede descuidarse un momento y causar perjuicios a las plantas, por lo menos se invertiría un salario mínimo vital, 354 dólares. Al manejarlo de forma automática el clima y riego se auto controlará, se ocupara mano de obra en forma parcial, solo para supervisión

Se estima que recuperaría la inversión de la implementación en un aproximado de dos meses, debido a que se invertiría 467 en la implementación, y dejaría de pagar sueldo de supervisión exhaustiva de 2 meses, los elementos automáticos harían todo el trabajo de control manual, además hay que tomar en cuenta la efectividad de los elementos electrónicos que lo controlan.

.

CONCLUSIONES:

- Al estudiar los elementos necesarios para la implementación del proyecto mediante tecnología de Arduino y Proccesing, se puede comprobar que esta tecnología es muy versátil al momento de desarrollar y trabajar con el lenguaje proporcionado en sus librerías, además de ser software y hardware libres.
- Con el diseño realizado en el presente proyecto sobre el sistema Arduino se ha logrado desarrollar un dispositivo de gran ayuda para el operador del cultivo, al mantener condiciones climáticas beneficiosas para los cultivos.
- Hay la facilidad de remplazar o agregar más funciones adicionales que le permita al sistema sea más robusto, seguro y versátil, al ser hardware y software libre puede ser modificado a cada necesidad
- El lenguaje Processing facilita interpretación de datos de entrada y salida de manera visual, siendo esta de fácil manejo para cualquier operador, solo se requerirá de conocimientos mínimos.

RECOMENDACIONES:

- En caso que se requiera agregar más funciones o adecuaciones en el dispositivo se pueden utilizar las ocho entradas análogas libres, y procesarlas de acuerdo a la necesidad.
- Se sugiere realizar un monitoreo periódico en los en los sensores, humedad excesiva en el sensor de temperatura, corrosión en la sonda de humedad del suelo, podrían afectar su funcionamiento normal.
- Se recomienda poner filtros en paralelo a las cargas inductivas, estas generan interferencias electromagnéticas en elementos electromecánicos presentes en los actuadores, pueden interferir como señal de entrada que no reconozca el Arduino,
- Se sugiere que los Arduino se dimensionen de acuerdo a la capacidad de procesamiento de señales, Arduinos de baja capacidad no interpretaran datos de acuerdo a lo requerido, pueden dar baja confiabilidad y fallas al sistema.

BIBLIOGRAFÍA:

Alpi, A., & Tognoni, F. . (1991). *Cultivo en invernadero.* Mundi-Prensa libros.

Arduino.cc. (29 de 02 de 2016). *Main Arduino.cc*.

Areny, R. (2004). *Sensores y Acondicionadores de Señal.* Marcombo.

Artero, O. T. (2013). *Arduino, Curso Practico de Formacion.* Madrid: Libros RC.

Autor, F. M. (2015). *Ecuador Patente nº 01.*

Cermeño, Z. S. (2005). *Construccion de Invernaderos.* Mundi-Prensa Libros.

Herias, FC. Gomez, G.G., Baeza, J:P., Bravo, C.J. (2015). Experiencias sobre el uso de la plataforma Arduino en Practicas de Automatizacion y Robotica. 84-101.

Herrador, R. (2009). *Guia de Usuario de Arduino.* Universidad de Cordova.

Hitek, P. (2015). *Panama Hitek*. Obtenido de www.panamahitek.com

hortalizas.com. (29 de 02 de 2016). *www. hortalizas.com*.

Infoagro. (2015). *www.infoagro.com*. Obtenido de www.infoagro.com/industria_auxiliar/tipo_invernaderos.htm

Leon, A., & Indra, W. (2002). *Redes de Comunicacion.* Madrid: Concepcion Fernandez Madrid.

LLEDO SANCHEZ, E. (2012). Diseño de un sistema de control domotico basado en la plataforma Arduino.

Microchip. (29 de 02 de 2016). *wws1.microchip.com*.

Perez, M., Alvarez, J., Campo, J., Ferrero, F., & Gustavo, G. (2004). *Instrumentacion Electronica.* España: Paraninfo.

Ponce, C. P. (2011). *www.hortalizas.com.*

Tanembaum, A. (2003). *Redes de Computadores.* Pearson Educacion.

ANEXOS:

Anexos 1. MANUAL DE USUARIO

INTRODUCCIÓN

Este manual describe la operación de todas las funciones básicas del prototipo de invernadero con control de temperatura y humedad. Desde la lectura de datos hasta el control automático y manual

Antes de proceder a la conexión del prototipo revise si están todos los componentes:

- Una fuente de poder
- Tarjeta de control con: Arduino Mega 2560, modulo Ethernet y router conectado, módulo de 8 relay
- Cables de conexión.
- Estructura de prototipo invernadero
- Actuadores: Foco de calefacción, ventilador, electro válvula, motor de cortina

Requisitos previos necesarios para usar el sistema.

Para utilizar el sistema debe estar instalado el programa Processing en su computador, el cual servirá de interfaz gráfica, además del programa XAMPP para cargar la base de datos de los cultivos, los dos programas son de descarga gratuita a cualquier momento.

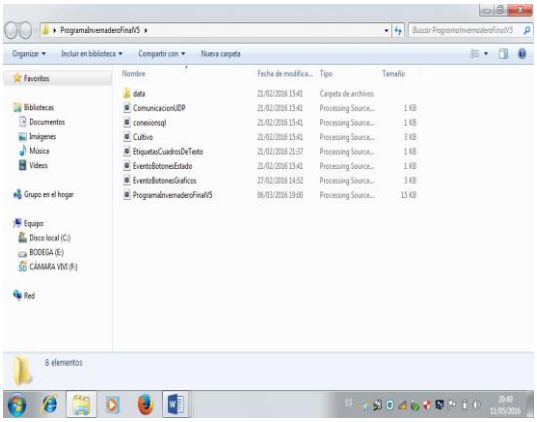
Instalación y configuración

- **Activación del modo manual**

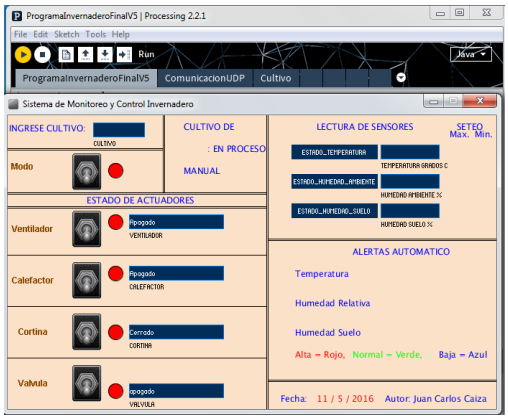
Para su instalación es necesario que se tenga previamente cargado el programa Processing, que servirá como interfaz gráfico, luego seguir los siguientes pasos:

1. Activar a ONN el interruptor de encendido ubicado en la parte frontal del tablero

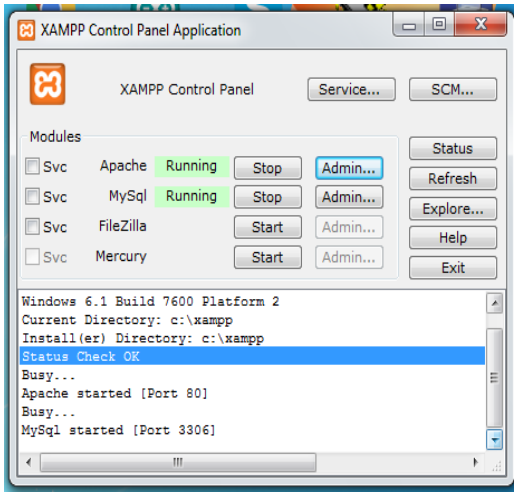
2. Establecer comunicación entre router y PC, identificando router Dlink
3. Abrir programa en Processing y ejecutarlo, el Archivo es ProgramaInvernaderoFinalV5.



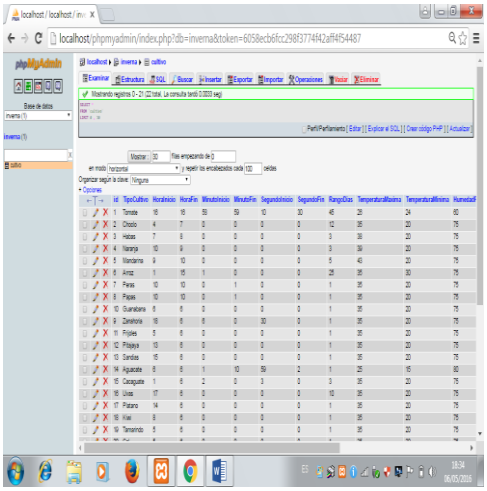
4. Luego aparecerá la interfaz gráfica, se puede activar en forma manual los actuadores para comprobar la existencia de conectividad.



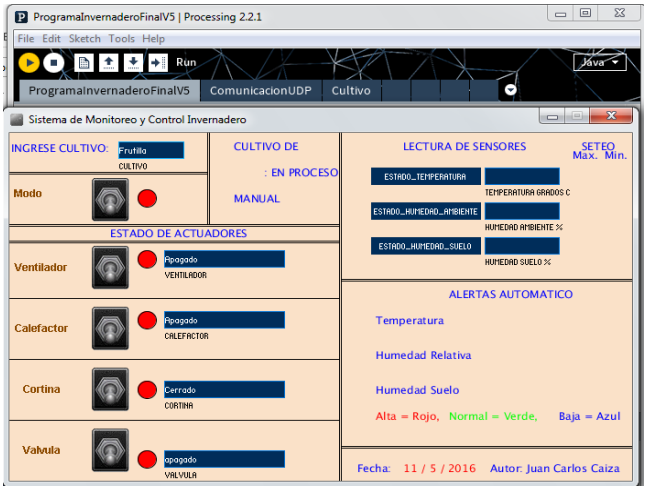
- **Activación del modo automático**
    1. Se deberá ejecutar los pasos del 1 al 4 del modo manual
    2. Abrir el programa XAMPP, ejecutar Apache y MySql, en el cual está la base de datos del cultivo.



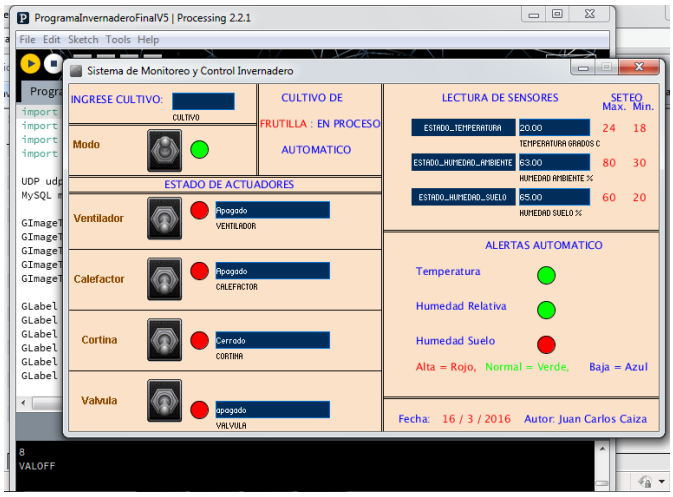
3 Si se desea ver base de datos, que está en la carpeta Inverna, incluir rangos de cultivo o editar ejecutar Admin del módulo Apache.

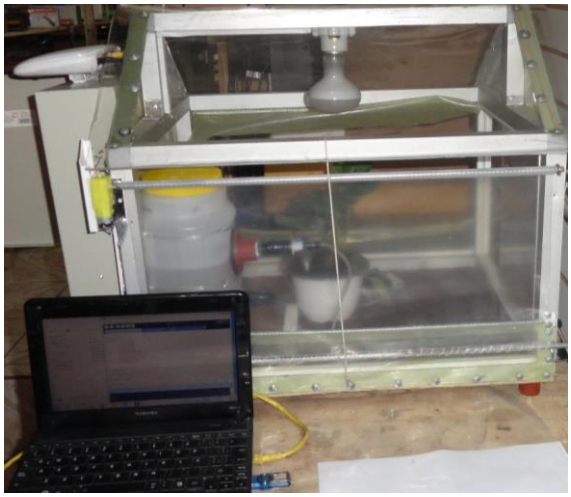


4 Cargar el cultivo que desea controlarse digitando en la casilla “Ingrese cultivo”, luego mover el interruptor (Modo), hacia arriba, el cual es el automático.



5 El prototipo empezara a monitorear parámetros de temperatura y humedad, comparar con valores programados, si se detectaran datos fuera de los parámetros, empezara a activar actuadores para estabilizar temperatura o humedad y mantenerla dentro del rango en el prototipo de invernadero.





Sección de solución de problemas

Se detalla los posibles errores o problemas más comunes que pueden surgir, junto con la forma de solucionarlos.

Falla	Solución
No enciende dispositivo	Verificar que alimentación general a tablero sea de 110 VAC
No envía datos dispositivo	Verificar que router este encendido y transmitiendo
No se encienden actuadores	Verificar en forma manual el funcionamiento de los actuadores, verificar si está cargado el cultivo en la aplicación
No se corrige baja temperatura	Verificar funcionamiento de foco de calefacción
No se corrige alta temperatura	Verificar funcionamiento de cortinas y ventilador
No se corrige humedad baja	Verificar funcionamiento de electroválvula y que depósito de agua tenga nivel suficiente

Dónde encontrar más ayuda, y datos de contacto.

Si el dispositivo no funciona correctamente acérquese a servicio técnico más adecuado

Anexos 2. PROGRAMACION DE BASE DE DATOS EN XAMP

En Processing se deberá importar la librería así:

importde.bezier.data.sql.*;

Generar un nuevo objeto para luego manejarlo:

MySQLmsql;

Cuando se tiene privilegios de administrador como clave y usuario se crean dos strings para poder ingresar estos textos como acceso:

String user = "root";

String pass = "admin";

Luego se crea otro string de datos para poder acceder al nombre de la base de datos:

Stringdatabase = "bildwelt";

Ahora está listos para hacer la conexión a la base de datos con los parámetros anteriores:

msql = new MySQL(this, "localhost", database, user, pass);

Todo esto lo hacen localmente porque solo es un sistema local en la computadora, para finalizar se usan los comandos de: a) si existe conexión y b) requerimiento de un contador para saber cuántos registros existen en nuestra base de datos:

```
if ( msql.connect() ) {

msql.query( "SELECT COUNT(*) FROM image" );

msql.next();

println( "number of rows: " + msql.getInt(1) );

}

else

 {

 // connectionfailed ! }
```

Se usaran los comandos para usar la base de datos:

"SELECT COUNT(*) FROM Cultivo"

Este comando es para enumerar cuantos elementos o filas contiene la tabla y poder seguir ingresando datos uno al final del otro

"Selectcount(id) as conteo from cultivo WhereTipoCultivo = 'Arroz'

Este commando sirve para buscar si existe un registro con el nombre Arroz y luego poder validar si crear o no ese tipo de cultivo

"SELECT HoraInicio FROM cultivos WHERE TipoCultivo='Arroz'"

Este comado sirve para buscar en el campo hora inicio del cultivo arroz y luego devolver la hora a la que se seteo este sistema

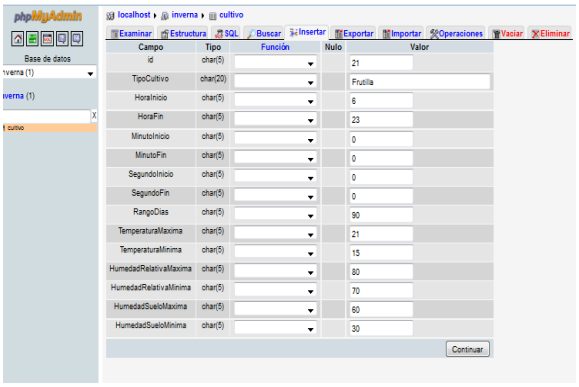
db.getInt("HoraInicio")

Obviamente se deben completar los pedidos a la base de datos con petición en forma de entero para saber que existe en ese campo

db.query("UPDATE cultivos SET "+SeteoCampo+" ="+valor+" WHERE TipoCultivo = "+"'"+TipoCultivo+"'"+" ");

Esta sentencia sirve para cambiar un dato de la tabla cultivo en donde en el campo seteo exista el tipo de cultivo como frutilla etc.

En la ilustracion, muestra valores programados para el cultivo a implementarse en este caso, frutilla.



Anexos 3. Programa implementado en Arduino y Processing

PROGRAMA EN ARDUINO MEGA

```
/*****************************************************************/
/*                  Incluyo Librerias                            */
/*****************************************************************/
#include "DHT.h"
#include <UIPEthernet.h>
#include <avr/wdt.h>
/*****************************************************************/
/*                  Defino Pines de Trabajo                      */
/*****************************************************************/
#define DHTPIN 2
#define DHTTYPE DHT11

#define FC1 3
#define FC2 4

#define IN1 5
#define IN2 6

#define CAL 7
#define VEN A5

#define AL1 A4
#define AL2 A3
#define AL3 A2

#define VAL A1


/*****************************************************************/
/*                  Inicializo Parametros                        */
/*****************************************************************/
DHT dht(DHTPIN, DHTTYPE);
EthernetUDP udp;

/*****************************************************************/
/*                   Variables Globales                          */
/*****************************************************************/
String mensajeUDP="";
String ackString="";
char charVal[7];
char charVal2[4];
float t;
float h;
float sensorHumedad=0;

/*****************************************************************/
/*           Funcion principal de configuraciones                */
/*****************************************************************/
void setup() {
  wdt_disable();
  Serial.begin(9600);/*Inicializo Comunicacion Serial a 9600 bps*/
  Serial.println("DHTxx test!");/*Imprimo Texto de Incio para la siguiente linea*/
  dht.begin();/*Inicializo la libreria del sensor de humedad y temperatura dht11*/
```

```
  uint8_t mac[6] = {0x00,0x01,0x02,0x03,0x04,0x05};
  Ethernet.begin(mac,IPAddress(192,168,0,106));
  int success = udp.begin(8000);
  Serial.print("initialize: ");
  Serial.println(success ? "success" : "failed");

  pinMode(FC1,INPUT);/*Fin de carrera para motor 1*/
  pinMode(FC2,INPUT);/*Fin de carrera para motor 2*/
  digitalWrite(FC1,HIGH);/*Activo la resistencia pull up FC1*/
  digitalWrite(FC2,HIGH);/*Activo la resistencia pull up FC2*/

  pinMode(IN1,OUTPUT);/*Pin de control Rele 1 para motor 1*/
  pinMode(IN2,OUTPUT);/*Pin de control Rele 2 para motor 2*/
  digitalWrite(IN1,LOW);/*Apago Rele 1*/
  digitalWrite(IN2,LOW);/*Apago Rele 2*/

   pinMode(CAL,OUTPUT);/*Pin de control Rele 3 para Calefactor*/
   digitalWrite(CAL,LOW);/*Apago Rele 3*/

   pinMode(VEN,OUTPUT);/*Pin de control Rele 4 para Ventilador*/
   digitalWrite(VEN,LOW);/*Apago Rele 4*/

   pinMode(AL1,OUTPUT);/*Pin de control Rele 5 para Alarma1*/
   pinMode(AL2,OUTPUT);/*Pin de control Rele 6 para Alarma2*/
   pinMode(AL3,OUTPUT);/*Pin de control Rele 7 para Alarma3*/
   digitalWrite(AL1,LOW);/*Apago Rele 5*/
   digitalWrite(AL2,LOW);/*Apago Rele 6*/
   digitalWrite(AL3,LOW);/*Apago Rele 7*/

   pinMode(VAL,OUTPUT);/*Pin de control Rele 8 para Valvula de Riego*/
   digitalWrite(VAL,LOW);/*Apago Rele 8*/

   //InicioCortina();//
   wdt_enable(WDTO_8S);
}

/*****************************************************************/
/*                  Lazo Principal de Trabajo                    */
/*****************************************************************/
void loop() {
  recepcionUDP();
  validacionUDP();
  wdt_reset();
}
++++++++++++++++++
int LecturaDHT11(String Sensor){
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t)) {
    Serial.println("Fallo al leer el sensor DHT11!");
    return 0;
  }

  if(Sensor=="humedad"){
    Serial.print("Humedad: ");
    Serial.print(h);
    Serial.println(" %");
```

Comentado [PS1]: Fuentes:
http://kaiwatechnology.com.ar/author/daniel/page/3/
http://kaiwatechnology.com.ar/tag/humedad/

```
    return h;
  }

  if(Sensor=="temperatura"){
    Serial.print("Temperatura: ");
    Serial.print(t);
    Serial.println(" *C");
    return t;
  }

}
++++++++++++++++++++++++++++++++++++++++++++
void MotorAbre(int tiempo){
  boolean est1 = digitalRead(FC1);
  boolean est2 = digitalRead(FC2);
  Serial.print("Estado1: ");Serial.println(est1);
  Serial.print("Estado2: ");Serial.println(est2);
  Serial.println(tiempo);

  while(digitalRead(FC1) == 0 && digitalRead(FC2) == 1){
    digitalWrite(IN1,HIGH);
    digitalWrite(IN2,LOW);
    delay(tiempo);
    //est1 = digitalRead(FC1);
    //est2 = digitalRead(FC2);
    while(digitalRead(FC1) == 1 && digitalRead(FC2) == 1){
      digitalWrite(IN1,HIGH);
      digitalWrite(IN2,LOW);
    //  est1 = digitalRead(FC1);
    //  est2 = digitalRead(FC2);
      delay(100);
    }
    delay(500);
    if(digitalRead(FC1) == 1 && digitalRead(FC2) == 0){
      Serial.println("Se Cerro");
    }
    if(digitalRead(FC1) == 0 && digitalRead(FC2) == 1){
      Serial.println("Se Abrio");
    }
    digitalWrite(IN1,LOW);
    digitalWrite(IN2,LOW);
  }
}

void MotorCierra(int tiempo){
  boolean est1 = digitalRead(FC1);
  boolean est2 = digitalRead(FC2);

  Serial.print("Estado1: ");Serial.println(est1);
  Serial.print("Estado2: ");Serial.println(est2);
  Serial.println(tiempo);

  while(digitalRead(FC1) == 1 && digitalRead(FC2) == 0){
    digitalWrite(IN2,HIGH);
    digitalWrite(IN1,LOW);
    delay(tiempo);
    est1 = digitalRead(FC1);
    est2 = digitalRead(FC2);
```

```
    while(digitalRead(FC1) == 1 && digitalRead(FC2) == 1){
      digitalWrite(IN2,HIGH);
      digitalWrite(IN1,LOW);
    //  est1 = digitalRead(FC1);
    //  est2 = digitalRead(FC2);
    delay(100);
    }
    delay(500);
    if(digitalRead(FC1) == 1 && digitalRead(FC2) == 0){
      Serial.println("Se Cerro");
    }
    if(digitalRead(FC1) == 0 && digitalRead(FC2) == 1){
      Serial.println("Se Abrio");
    }
    digitalWrite(IN2,LOW);
    digitalWrite(IN1,LOW);
  }
}
++++++++++++++++++++++++++++++
int SensorHumedadSuelo(){
  float sh = 0.00;
  unsigned long valoradc = 0;

  for(int x = 1 ; x < 101 ; x++){
    valoradc += analogRead(A0);
  }

  valoradc = valoradc/100;
  sh = (valoradc*5.00)/1024.00;

  //Serial.print("ADC: ");Serial.println(valoradc);
  //Serial.print("SH: ");Serial.println(sh);

  valoradc = map(valoradc,0,1024,100,0);
  Serial.print("ADC: ");Serial.print(valoradc);Serial.println(" %");
  return valoradc;
}
++++++++++++++++++++++++++++++++
void InicioCortina(){
  Serial.print("Estado fin de carrera 1"); Serial.println(digitalRead(FC1));
  Serial.print("Estado fin de carrera 2"); Serial.println(digitalRead(FC2));
  if(digitalRead(FC1) == 0 && digitalRead(FC2) == 0){
    Serial.println("Sistema Trabado por favor revisar");
  }
  if(digitalRead(FC1) == 1 && digitalRead(FC2) == 0){
    digitalWrite(IN2,HIGH);
    digitalWrite(IN1,LOW);
    while(digitalRead(FC1) == 1 && digitalRead(FC2) == 1){
      digitalWrite(IN1,HIGH);
      digitalWrite(IN2,LOW);
    }
    digitalWrite(IN2,LOW);
    digitalWrite(IN1,LOW);
    Serial.println("Fin de Proceso 1");//abrir
  }
  if(digitalRead(FC1) == 0 && digitalRead(FC2) == 1){
    digitalWrite(IN2,LOW);
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
void recepcionUDP(){
  int size = udp.parsePacket();
  if (size > 0) {
    do
     {
       char* msg = (char*)malloc(size+1);
       int len = udp.read(msg,size+1);
       msg[len]=0;
       mensajeUDP+=msg;
       Serial.println(msg);
       free(msg);
     }
    while ((size = udp.available())>0);
    udp.flush();
    int success;
    do{
      success = udp.beginPacket(udp.remoteIP(),udp.remotePort());
    }
    while (!success);
    success = udp.println(ackString);
    success = udp.endPacket();
    udp.stop();
    udp.begin(8000);
    ackString="";
  }
}

void validacionUDP(){
  if(mensajeUDP != ""){
    Serial.println(mensajeUDP);
    mensajeUDP.trim();

    if(mensajeUDP == "temperatura"){
      Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
      mensajeUDP="";
      ackString+="t";
      float vtemp = LecturaDHT11("temperatura");
      dtostrf(vtemp, 3, 2, charVal);
      for(int i=0;i<sizeof(charVal);i++)
      {
        ackString+=charVal[i];
      }
      ackString.trim();
    }

    if(mensajeUDP == "humedadr"){
      Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
      mensajeUDP="";
      ackString+="hr";
      float hrtemp = LecturaDHT11("humedad");
      dtostrf(hrtemp, 3, 2, charVal);
      for(int i=0;i<sizeof(charVal);i++)
      {
        ackString+=charVal[i];
      }
      ackString.trim();
    }
```

Comentado [PS2]: Cita, fuentes:
https://uvadoc.uva.es/bitstream/10324/5863/1/PFC-B.16.pdf
http://kaiwatechnology.com.ar/author/daniel/page/3/
http://www.taringa.net/posts/hazlo-tu-mismo/14038589/Arduino-que-es.html
https://ieszoco-tecnologia.wikispaces.com/file/view/ARDUINO nivel 1.pdf
http://myslide.es/documents/hack-x-crackcuadernoarduino.html

```
if(mensajeUDP == "humedads"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  ackString+="hs";
  float hstemp = SensorHumedadSuelo();
  dtostrf(hstemp, 3, 2, charVal);
  for(int i=0;i<sizeof(charVal);i++)
  {
    ackString+=charVal[i];
  }
  ackString.trim();
}

if(mensajeUDP == "alarma1"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  boolean a1temp = digitalRead(AL1);
  dtostrf(a1temp, 3, 2, charVal);
  for(int i=0;i<sizeof(charVal);i++)
  {
    ackString+=charVal[i];
  }
  ackString.trim();
}

if(mensajeUDP == "alarma2"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  boolean a2temp = digitalRead(AL2);
  dtostrf(a2temp, 3, 2, charVal);
  for(int i=0;i<sizeof(charVal);i++)
  {
    ackString+=charVal[i];
  }
  ackString.trim();
}

if(mensajeUDP == "alarma3"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  boolean a3temp = digitalRead(AL3);
  dtostrf(a3temp, 3, 2, charVal);
  for(int i=0;i<sizeof(charVal);i++)
  {
    ackString+=charVal[i];
  }
  ackString.trim();
}

if(mensajeUDP == "alarma1on"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  digitalWrite(AL1,HIGH);
  ackString="AL1ON";
  ackString.trim();
}

if(mensajeUDP == "alarma2on"){
```

```
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  digitalWrite(AL2,HIGH);
  ackString="AL2ON";
  ackString.trim();
}

if(mensajeUDP == "alarma3on"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  digitalWrite(AL3,HIGH);
  ackString="AL3ON";
  ackString.trim();
}

if(mensajeUDP == "alarma1off"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  digitalWrite(AL1,LOW);
  ackString="AL1OFF";
  ackString.trim();
}

if(mensajeUDP == "alarma2off"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  digitalWrite(AL2,LOW);
  ackString="AL2OFF";
  ackString.trim();
}

if(mensajeUDP == "alarma3off"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  digitalWrite(AL3,LOW);
  ackString="AL3OFF";
  ackString.trim();
}

if(mensajeUDP == "calefactoron"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  digitalWrite(CAL,HIGH);
  ackString="CALON";
  ackString.trim();
}

if(mensajeUDP == "calefactoroff"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
  digitalWrite(CAL,LOW);
  ackString="CALOFF";
  ackString.trim();
}

if(mensajeUDP == "ventiladoron"){
  Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
  mensajeUDP="";
```

```
    digitalWrite(VEN,HIGH);
    ackString="VENON";
    ackString.trim();
   }

   if(mensajeUDP == "ventiladoroff"){
    Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
    mensajeUDP="";
    digitalWrite(VEN,LOW);
    ackString="VENOFF";
    ackString.trim();
   }

   if(mensajeUDP == "valvulaon"){
    Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
    mensajeUDP="";
    digitalWrite(VAL,HIGH);
    ackString="VALON";
    ackString.trim();
   }

   if(mensajeUDP == "valvulaoff"){
    Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
    mensajeUDP="";
    digitalWrite(VAL,LOW);
    ackString="VALOFF";
    ackString.trim();
   }

   Serial.println(mensajeUDP.substring(0,6));
   if(mensajeUDP.substring(0,6) == "motord"){
    String valor = mensajeUDP.substring(6);
    Serial.println(valor);
    Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
    int eval = valor.toInt();
    mensajeUDP="";
    MotorAbre(eval);
    ackString="MOTDOK";
    ackString.trim();
   }

   if(mensajeUDP.substring(0,6) == "motori"){
    String valor = mensajeUDP.substring(6);
    Serial.println(valor);
    Serial.print("Peticion ");Serial.print(mensajeUDP);Serial.println(" Aceptado");
    int eval = valor.toInt();
    mensajeUDP="";
    MotorCierra(eval);
    ackString="MOTIOK";
    ackString.trim();
   }

   mensajeUDP="";//libero espacio si no existen comando detectados
  }
}
void Calefactor(String estado){
  if(estado == "ON"){
    digitalWrite(CAL,HIGH);
```

```
  }

  if(estado == "OFF"){
    digitalWrite(CAL,LOW);
  }

}

void Ventilacion(String estado){
  if(estado == "ON"){
    digitalWrite(VEN,HIGH);
  }

  if(estado == "OFF"){
    digitalWrite(VEN,LOW);
  }

}

void Alarma1(String estado){
  if(estado == "ON"){
    digitalWrite(AL1,HIGH);
  }

  if(estado == "OFF"){
    digitalWrite(AL1,LOW);
  }

}

void Alarma2(String estado){
  if(estado == "ON"){
    digitalWrite(AL2,HIGH);
  }

  if(estado == "OFF"){
    digitalWrite(AL2,LOW);
  }

}

void Alarma3(String estado){
  if(estado == "ON"){
    digitalWrite(AL3,HIGH);
  }

  if(estado == "OFF"){
    digitalWrite(AL3,LOW);
  }

}

void Valvula(String estado){
  if(estado == "ON"){
    digitalWrite(VAL,HIGH);
  }

  if(estado == "OFF"){
```

```
    digitalWrite(VAL,LOW);
  }

}

  //LecturaDHT11("humedad");
  //LecturaDHT11("temperatura");
  //SensorHumedadSuelo();
  //MotorAbre(500);
  //MotorCierra(500);
  //Calefactor("ON");
  //delay(500);
  //Calefactor("OFF");
  //Ventilacion("ON");
  //delay(500);
  //Ventilacion("OFF");
  //Alarma1("ON");
  //delay(250);
  //Alarma2("ON");
  //delay(250);
  //Alarma3("ON");
  //delay(250);
  //Alarma1("OFF");
  //Alarma2("OFF");
  //Alarma3("OFF");
  //Valvula("ON");
  //delay(100);
  //Valvula("OFF");
```

PROGRAMA PROCESSING

```
import g4p_controls.*;
import controlP5.*;
import hypermedia.net.*;
import de.bezier.data.sql.*;

UDP udp;
MySQL msql;

GImageToggleButton btn1;
GImageToggleButton btn2;
GImageToggleButton btn3;
GImageToggleButton btn4;
GImageToggleButton btn5;

GLabel lbl1;
GLabel lbl2;
GLabel lbl3;
GLabel lbl4;
GLabel lbl5;
GLabel lbl6;

ControlP5 cp5;
Textfield myTextfield1,
        myTextfield2,
        myTextfield3,
        myTextfield4,
        myTextfield5,
```

```
        EstadoTemperatura,
        EstadoHumedadAmbiente,
        EstadoHumedadSuelo,
        EstadoCortina,
        Estadovalvula
        ;

byte cont=0;
int cont1=0;
String inputString="";
String ip = "192.168.0.106";  // the remote IP address
int port  = 8000;    // the destination port

String modo="";
String temp="";
String hume="";
String mot="";

String etemp="";
String ehum1="";
String ehum2="";
String ecort="";
String evalv="";

int m=0;
int fmot =0;
int ftemp=0;
int fhum =0;
int fval =0;
int fhum2=0;
int fmod=0;

int offsetX=60;
int offsetY=80;

float n,n1;
int flag_inicio_botones=0;

String numero="";

int dia;    // Values from 1 - 31
int mes;  // Values from 1 - 12
int anio;   // 2003, 2004, 2005, etc.
String fecha;

String user     = "root";
String pass     = "1234";
String database = "Inverna";
String CualCultivo = "";
String NombresCultivos="";
String[] list;
int aux1;

int RangoDias;

int HoraInicio;
int MinutoInicio;
int SegundoInicio;
```

```
int HoraFin;
int MinutoFin;
int SegundoFin;

int TemperaturaMaxima;
int TemperaturaMinima;

int HumedadRelativaMaxima;
int HumedadRelativaMinima;

int HumedadSueloMaxima;
int HumedadSueloMinima;

int segundos = second();  // Values from 0 - 59
int minutos = minute();  // Values from 0 - 59
int horas = hour();    // Values from 0 - 23

float tempActual;
float HumRelActual;
float HumSueActual;

int contaux=0;
boolean flagapagado=false;
boolean flagTC=false;
String nombreCultivo="";


/*int VARIABLE_TIEMPO_DERECHA=1000;*/

public void setup() {
  size(680,400, JAVA2D);
  PFont font = createFont("arial",20);
  G4P.setGlobalColorScheme(GCScheme.ORANGE_SCHEME);
  G4P.setCursor(ARROW);
  if (frame != null)
    frame.setTitle("Sistema de Monitoreo y Control Invernadero");
  // Create image toggle buttons
  btn1 = new GImageToggleButton(this, 10+80, 10+42);
  btn1.tag = "button1 ";
  btn2 = new GImageToggleButton(this, 10+80, 80+50);
  btn2.tag = "button2 ";
  btn3 = new GImageToggleButton(this, 10+80, 150+50);
  btn3.tag = "button3 ";
  btn4 = new GImageToggleButton(this, 10+80, 220+50);
  btn4.tag = "button4 ";
  btn5 = new GImageToggleButton(this, 10+80, 290+50);
  btn5.tag = "button5 ";

  createLabels();

  cp5 = new ControlP5(this);

  myTextfield2 = cp5.addTextfield("Cultivo"        ,200-80,25-15,70,20).setColorLabel (color(0));
  myTextfield1 = cp5.addTextfield("Ventilador" ,200-30,95+40,130,20).setColorLabel (color(0));
  myTextfield3 = cp5.addTextfield("Calefactor"     ,200-30,165+40,130,20).setColorLabel(color(0));
  myTextfield4 = cp5.addTextfield("Cortina"  ,200-30,245+40,130,20).setColorLabel(color(0));
  myTextfield5 = cp5.addTextfield("Valvula"  ,200-30,325+40,130,20).setColorLabel(color(0));
```

```
  EstadoTemperatura = cp5.addTextfield("Temperatura Grados C"  ,520,150-110,80,20).setColorLabel(color(0));
  EstadoHumedadAmbiente = cp5.addTextfield("Humedad Ambiente %"  ,520,190-
110,80,20).setColorLabel(color(0));
  EstadoHumedadSuelo = cp5.addTextfield("Humedad Suelo %"  ,520,230-110,80,20).setColorLabel(color(0));

  myTextfield1.setFocus(false);
  myTextfield3.setFocus(false);
  myTextfield4.setFocus(false);
  myTextfield5.setFocus(false);



   EstadoTemperatura.setFocus(false);
   EstadoHumedadAmbiente.setFocus(false);
   EstadoHumedadSuelo.setFocus(false);

  // create a new button with name 'buttonA'
  cp5.addButton("Estado_Temperatura")
     .setValue(0)
     .setPosition(335+offsetX,120-offsetY)
     .setSize(120,20)
     ;

  // and add another 2 buttons
  cp5.addButton("Estado_Humedad_Ambiente")
     .setValue(100)
     .setPosition(335+offsetX,160-offsetY)
     .setSize(120,20)
     ;

  cp5.addButton("Estado_Humedad_Suelo")
     .setPosition(335+offsetX,200-offsetY)
     .setSize(120,20)
     .setValue(100)
     ;

  udp = new UDP( this, 2000 );
  udp.listen( true );

  dia = day();    // Values from 1 - 31
  mes = month();  // Values from 1 - 12
  anio = year();   // 2003, 2004, 2005, etc.
  msql = new MySQL( this, "localhost", database, user, pass );
  PrimeraConexion();

}

public void draw() {
  background(250, 225, 200);

  fecha = String.valueOf(dia);
  text(fecha+" /", 400+30, 300+90);
  fecha = String.valueOf(mes);
  text(" "+fecha+" / ", 425+30, 300+90);
  fecha = String.valueOf(anio);
  text(" "+fecha, 445+30, 300+90);
  segundos = second();  // Values from 0 - 59
```

```
  minutos = minute();  // Values from 0 - 59
  horas = hour();    // Values from 0 - 23
  if(flagTC == true)
  if(fmod==1){text(nombreCultivo,220,50);
  text(TemperaturaMaxima, 615,55);
text(TemperaturaMinima, 650,55);
text(HumedadRelativaMaxima, 615,95);
text(HumedadRelativaMinima, 650,95);
text(HumedadSueloMaxima, 615,135);
text(HumedadSueloMinima, 650,135);

}
  fill(255,0,0);
  text("Alta = Rojo,",400,350-20);
  fill(0,255,0);
  text("Normal = Verde,",400+80,350-20);
  fill(0,0,255);
  text("Baja = Azul",400+200,350-20);
 text("INGRESE CULTIVO:",02 ,22);
 text("CULTIVO DE", 245, 20);
 text(": EN PROCESO",278,50);
 text("ESTADO DE ACTUADORES",110,120);
 text("LECTURA DE SENSORES",430,20);
 text("SETEO",625,20);
 text("Max.", 615,30 );
 text("Min.", 650,30);
 text("ALERTAS AUTOMATICO",480,190);
 text("Temperatura",400,220);
 text("Humedad Relativa",400,260);
 text("Humedad Suelo",400,300);
 fill(0,0,255);

 line(390-30,00,390-30,450);
 line(392-30,0,392-30,450);
 line(0,105,390-30,105);
 line(218,0,218,105);
 line(216,0,216,105);
 line(362,170,800,170);
 line(362,168,800,168);
 line(0,107,390-30,107);
 line(0,45,216,45);
 line(0,47,216,47);

line(0,124,360,124);
line(0,126,360,126);

line(0,190,360,190);
line(0,260,360,260);
line(0,330,360,330);

line(362,360,800,360);
line(362,362,800,362);

  cont++;
  myTextfield1.setText(temp);
  myTextfield3.setText(hume);
  myTextfield4.setText(hume);
  myTextfield5.setText(mot);
```

```
EstadoTemperatura.setText(etemp);
EstadoHumedadAmbiente.setText(ehum1);
EstadoHumedadSuelo.setText(ehum2);
fill(0,0,255);
text("Fecha:",380,390);
text("Autor: Juan Carlos Caiza",525,390);


if(fmod==0){
  text("MANUAL",245,40+40);
  fill(255,0,0);

}
if(fmod==1){
  text("AUTOMATICO",245,40+40);
  fill(0,255,0);

}
ellipse(350-200, 35+40, 20, 20);

if(ftemp==0){
  fill(255,0,0);
  myTextfield1.setText("Apagado");
}
if(ftemp==1){
  fill(0,255,0);
  myTextfield1.setText("Encendido");
}
ellipse(350-200, 105+40, 20, 20);

if(fhum==0){
  fill(255,0,0);
   myTextfield3.setText("Apagado");//<----------
}
if(fhum==1){
  fill(0,255,0);
  myTextfield3.setText("Encendido");
}
ellipse(350-200, 175+40, 20, 20);

if(fhum2==0){
  fill(255,0,0);
  myTextfield4.setText("Cerrado");
}
if(fhum2==1){
  fill(0,255,0);
  myTextfield4.setText("Abierto");
}
ellipse(350-200, 255+40, 20, 20);

if(fval==0){
  fill(255,0,0);
  myTextfield5.setText("apagado");
}
if(fval==1){
  fill(0,255,0);
  myTextfield5.setText("encendido");
```

```
}
ellipse(350-200, 335+40, 20, 20);

if(cont>50){flag_inicio_botones=1;}



//modo automatico dentro del siguiente if
if(m==1){
  segundos = second();  // Values from 0 - 59

  if(segundos >= 1 && segundos <= 5) {
    flag_inicio_botones=1;
    udp.send("temperatura", ip, port );
    delay(50);
  }

  if(segundos >= 21 && segundos <= 25) {
    flag_inicio_botones=1;
    udp.send("humedadr", ip, port );
    delay(50);
  }

  if(segundos >= 45 && segundos <= 59) {
    flag_inicio_botones=1;
    udp.send("humedads", ip, port );
    delay(50);
    println("acabo....");
  }


  //para comentar y testear rapido
  if(horas == 23 && minutos == 59 && segundos == 59){
    println("Fin de la Jornada");
    while(horas == 23 && minutos == 59 && segundos == 59){}

    RangoDias--;

  }

  RangoDias=1;
  if(RangoDias > 0)


  {

    //tempActual = 25; TemperaturaMaxima=26;TemperaturaMinima=24;
    if(tempActual > TemperaturaMaxima && tempActual > 0){
      fill(255,0,0);
      ellipse(550, 220, 20, 20);
      udp.send("calefactoroff"+char(13)+char(10), ip, port );   // the message to send
      delay(250);
      udp.send("motord3000"+char(13)+char(10), ip, port );   // the message to send
      delay(250);
      udp.send("alarma1on"+char(13)+char(10), ip, port );   // the message to send
      delay(250);
      udp.send("ventiladoron"+char(13)+char(10), ip, port );   // the message to send
      delay(250);
```

```
  ftemp=1;//ventilador
  fhum=0;//calefactor
  fhum2=1;//cortina
  fval=0;//valvula
 }else
 if(tempActual < TemperaturaMinima && tempActual > 0 ){
  fill(0,0,255);
  ellipse(550, 220, 20, 20);
  udp.send("motori3000"+char(13)+char(10), ip, port );
  delay(250);
  udp.send("calefactoron"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
  udp.send("alarma1on"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
  udp.send("ventiladoroff"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
  ftemp=0;//ventilador
  fhum=1;//calefactor
  fhum2=0;//cortina
  fval=0;//valvula
 }else
 if(tempActual >= TemperaturaMinima && tempActual <= TemperaturaMaxima && tempActual > 0){
  fill(0,255,0);
  ellipse(550, 220, 20, 20);
  udp.send("motori3000"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
  udp.send("calefactoroff"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
  udp.send("alarma1off"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
  udp.send("ventiladoroff"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
  ftemp=0;//ventilador
  fhum=0;//calefactor
  fhum2=0;//cortina
  fval=0;//valvula
 }

 //HumRelActual=40;HumedadRelativaMaxima=60;HumedadRelativaMinima=50;
 if(HumRelActual > HumedadRelativaMaxima && HumRelActual>0){
  fill(255,0,0);
  ellipse(550, 260, 20, 20);
   udp.send("alarma2on"+char(13)+char(10), ip, port );   // the message to send
   delay(250);

 }else
 if(HumRelActual < HumedadRelativaMinima && HumRelActual>0){
  fill(0,0,255);
  ellipse(550, 260, 20, 20);
  udp.send("alarma2on"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
 }else
 if(HumRelActual >= HumedadRelativaMinima && HumRelActual <= HumedadRelativaMaxima &&
HumRelActual>0){
  fill(0,255,0);
  ellipse(550, 260, 20, 20);
  udp.send("alarma2off"+char(13)+char(10), ip, port );   // the message to send
  delay(250);
```

```
    }


    //HumSueActual = 40;HumedadSueloMaxima=70;HumedadSueloMinima=50;
    if(HumSueActual > HumedadSueloMaxima && HumSueActual>0){
      fill(255,0,0);
      ellipse(550, 300, 20, 20);
      //text("alto",550,350);
      udp.send("alarma3on"+char(13)+char(10), ip, port );   // the message to send
      delay(250);

      udp.send("valvulaoff", ip, port );   // the message to send
      delay(250);
      //ftemp=0;//ventilador
      //fhum=0;//calefactor
      //fhum2=0;//cortina
      fval=0;//valvula

    }else
    if(HumSueActual < HumedadSueloMinima && HumSueActual>0){
      fill(0,0,255);
      ellipse(550, 300, 20, 20);
      udp.send("alarma3on"+char(13)+char(10), ip, port );   // the message to send
      delay(250);
      //**************************************************************************+
      //if(horas>HoraInicio && horas>HoraFin){
      //  if(minutos>MinutoInicio && minutos>MinutoFin){
          if(segundos>SegundoInicio && segundos>SegundoFin){
            udp.send("valvulaon", ip, port );   // the message to send
            delay(250);
            //ftemp=0;//ventilador
            //fhum=0;//calefactor
            //fhum2=0;//cortina
            fval=1;//valvula
          }else{
            udp.send("valvulaoff", ip, port );   // the message to send
            delay(250);
            //ftemp=0;//ventilador
            //fhum=0;//calefactor
            //fhum2=0;//cortina
            fval=0;//valvula
          }
      //  }
      //}
      //**************************************************************************+

    }else
    if(HumSueActual >= HumedadSueloMinima && HumSueActual <= HumedadSueloMaxima &&
HumSueActual>0){
      fill(0,255,0);
      ellipse(550, 300, 20, 20);
      udp.send("alarma3off"+char(13)+char(10), ip, port );   // the message to send
      delay(250);

      udp.send("valvulaoff", ip, port );   // the message to send
      delay(250);
      //ftemp=0;//ventilador
      //fhum=0;//calefactor
```

```
      //fhum2=0;//cortina
      fval=0;//valvula

    }

   flagapagado=true;
  }
 }else{
     if(flagapagado == true){
       int x=0;
       for(x=0;x<5;x++){
         udp.send("alarma1off"+char(13)+char(10), ip, port );
         delay(100);
         udp.send("alarma2off"+char(13)+char(10), ip, port );
         delay(100);
         udp.send("alarma3off"+char(13)+char(10), ip, port );
         delay(100);
         udp.send("motori5000"+char(13)+char(10), ip, port );   // cierra cortina
         delay(100);
         udp.send("calefactoroff"+char(13)+char(10), ip, port );   // the message to send
         delay(100);
         udp.send("valvulaoff"+char(13)+char(10), ip, port );   // the message to send
         delay(100);
         udp.send("ventiladoroff"+char(13)+char(10), ip, port );   // the message to send
         delay(100);
       }
       flagapagado=false;
     }
 }



++++++++++++++++++++++
void receive( byte[] data ) {
 for(int i=0; i < data.length; i++){
   inputString+=char(data[i]);
 }

 inputString.trim();
 println(inputString.length());
 print(inputString);
 println();

 if(inputString.substring(0,1).equals("t")){
   println("llego la temperatura");
   etemp=inputString.substring(1).toString();
   tempActual=float(etemp);
 }

 if(inputString.substring(0,2).equals("hr")){
   println("llego la humedad relativa");
   ehum1=inputString.substring(2).toString();
   HumRelActual=float(ehum1);
 }

 if(inputString.substring(0,2).equals("hs")){
   println("llego la humedad del suelo");
   ehum2=inputString.substring(2).toString();
```

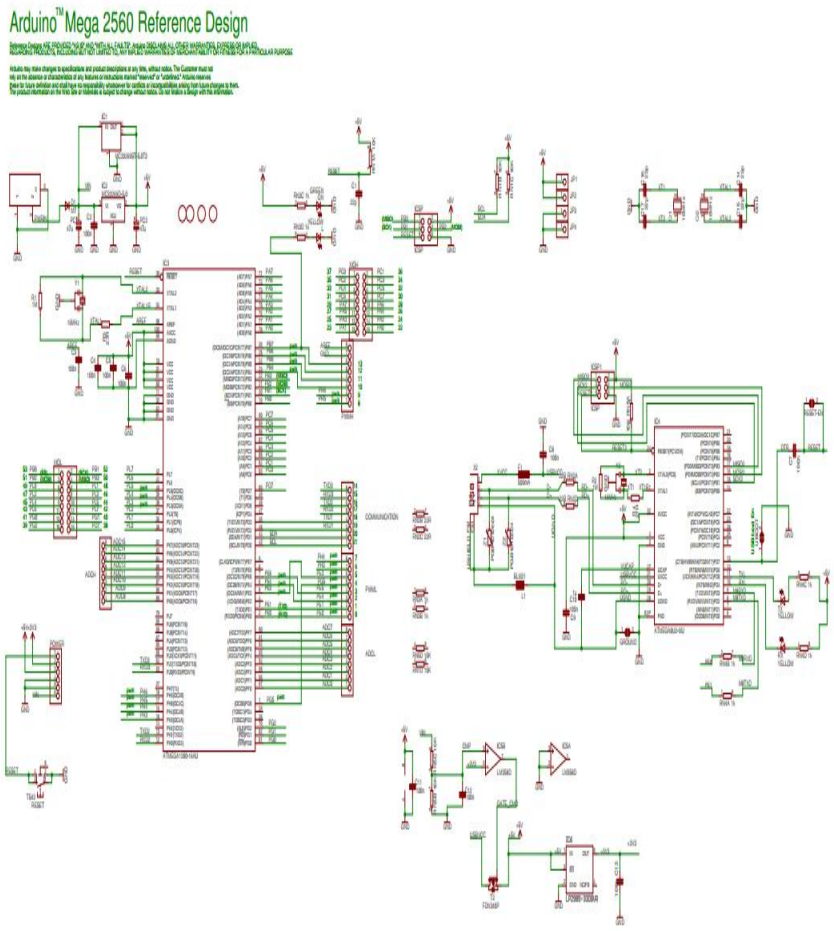
```
   HumSueActual=float(ehum2);
 }

 inputString="";
}
}
public void Cultivo(String theText) {
  // automatically receives results from controller input
  println("a textfield event for controller 'Modo' : "+theText);
  flagTC=true;
  nombreCultivo=theText;
  msql.query( "SELECT RangoDias FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  RangoDias=msql.getInt("RangoDias");println("RangoDias: "+RangoDias);

  msql.query( "SELECT HoraInicio FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  HoraInicio=msql.getInt("HoraInicio");println("HoraInicio: "+HoraInicio);

  msql.query( "SELECT MinutoInicio FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  MinutoInicio=msql.getInt("MinutoInicio");println("MinutoInicio: "+MinutoInicio);

  msql.query( "SELECT SegundoInicio FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  SegundoInicio=msql.getInt("SegundoInicio");println("SegundoInicio: "+SegundoInicio);

  msql.query( "SELECT HoraFin FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  HoraFin=msql.getInt("HoraFin");println("HoraFin: "+HoraFin);

  msql.query( "SELECT MinutoFin FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  MinutoFin=msql.getInt("MinutoFin");println("MinutoFin: "+MinutoFin);

  msql.query( "SELECT SegundoFin FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  SegundoFin=msql.getInt("SegundoFin");println("SegundoFin: "+SegundoFin);

  msql.query( "SELECT TemperaturaMaxima FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  TemperaturaMaxima=msql.getInt("TemperaturaMaxima");println("TemperaturaMaxima:
"+TemperaturaMaxima);

  msql.query( "SELECT TemperaturaMinima FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  TemperaturaMinima=msql.getInt("TemperaturaMinima");println("TemperaturaMinima: "+TemperaturaMinima);

  msql.query( "SELECT HumedadRelativaMaxima FROM cultivo where TipoCultivo = '"+theText+"'"
);msql.next();
  HumedadRelativaMaxima=msql.getInt("HumedadRelativaMaxima");println("HumedadRelativaMaxima:
"+HumedadRelativaMaxima);

  msql.query( "SELECT HumedadRelativaMinima FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  HumedadRelativaMinima=msql.getInt("HumedadRelativaMinima");println("HumedadRelativaMinima:
"+HumedadRelativaMinima);

  msql.query( "SELECT HumedadSueloMaxima FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  HumedadSueloMaxima=msql.getInt("HumedadSueloMaxima");println("HumedadSueloMaxima:
"+HumedadSueloMaxima);

  msql.query( "SELECT HumedadSueloMinima FROM cultivo where TipoCultivo = '"+theText+"'" );msql.next();
  HumedadSueloMinima=msql.getInt("HumedadSueloMinima");println("HumedadSueloMinima:
"+HumedadSueloMinima);

}
```

```
public void createLabels() {
  lbl1 = new GLabel(this,0, 10+40, 40, 40);
  lbl1.setText("Modo");
  lbl1.setTextBold();
  lbl1.setOpaque(true);

  lbl2 = new GLabel(this, 60-60, 80+50, 70, 50);
  lbl2.setText("Ventilador");
  lbl2.setTextBold();
  lbl2.setOpaque(true);

  lbl3 = new GLabel(this, 60-60, 150+50, 70, 50);
  lbl3.setText("Calefactor");
  lbl3.setTextBold();
  lbl3.setOpaque(true);

  lbl4 = new GLabel(this, 60-60, 220+50, 70, 50);
  lbl4.setText("Cortina");
  lbl4.setTextBold();
  lbl4.setOpaque(true);



  lbl5 = new GLabel(this, 60-60, 290+50, 70, 50);
  lbl5.setText("Valvula");
  lbl5.setTextBold();
  lbl5.setOpaque(true);

}
public void controlEvent(ControlEvent theEvent) {
  println(theEvent.getController().getName());

  if(flag_inicio_botones==1){
    if(theEvent.getController().getName().equals("Estado_Temperatura")){
      println("temp");
      udp.send("temperatura", ip, port );
    }
    if(theEvent.getController().getName().equals("Estado_Humedad_Ambiente")){
      println("hum1");
      udp.send("humedadr", ip, port );
    }
    if(theEvent.getController().getName().equals("Estado_Humedad_Suelo")){
      println("hum2");
      udp.send("humedads", ip, port );
    }
    /*if(theEvent.getController().getName().equals("Estado_Motor_Cortina")){
      println("cort");
      //udp.send("stateCort", ip, port );
    }
    if(theEvent.getController().getName().equals("Estado_Valvula_Riego")){
      println("valv");
     //udp.send("stateValv", ip, port );
    }*/
  }
}
// Event handler for image toggle buttons
public void handleToggleButtonEvents(GImageToggleButton button,GEvent event) {
  //println(button + "State: " + button.getState());
```

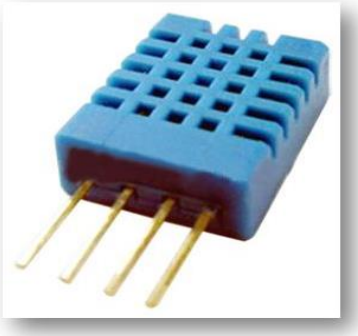
```
  String aux = button.toString() ;

  if(aux == "button1 "){
    int a=btn1.getState();
    if(a==0){
      println("MOD0: "+btn1.getState());
      //udp.send("MOD0", ip, port );   // the message to send
      m=0;
      fmod=a;
    }
    if(a==1){
      println("MOD1: "+btn1.getState());
      //udp.send("MOD1", ip, port );   // the message to send
      m=1;
      fmod=a;
    }
  }
if(m==0){
  if(aux == "button2 "){
    int b=btn2.getState();
    if(b==0){
      println("ventiladoroff: "+btn2.getState());
      udp.send("ventiladoroff"+char(13)+char(10), ip, port );   // the message to send
      ftemp=b;
    }
    if(b==1){
      println("ventiladoron: "+btn2.getState());
      udp.send("ventiladoron"+char(13)+char(10), ip, port );   // the message to send
      ftemp=b;
    }
  }

  if(aux == "button3 "){
    int c=btn3.getState();
    if(c==0){
      println("calefactoroff: "+btn3.getState());
      udp.send("calefactoroff"+char(13)+char(10), ip, port );   // the message to send
      fhum=c;
    }
    if(c==1){
      println("calefactoron: "+btn3.getState());
      udp.send("calefactoron"+char(13)+char(10), ip, port );   // the message to send
      fhum=c;
    }
  }

  if(aux == "button4 "){
    int d=btn4.getState();
    if(d==0){
      println("motori3000: "+btn4.getState());
      udp.send("motori3000"+char(13)+char(10), ip, port );   // the message to send
      fhum2=d;
    }
    if(d==1){
      println("motord3000: "+btn4.getState());
      udp.send("motord3000"+char(13)+char(10), ip, port );   // the message to send
      fhum2=d;
    }
```

```
  }

  if(aux == "button5 "){
    int e=btn5.getState();
    if(e==0){
      println("valvulaoff: "+btn5.getState());
      udp.send("valvulaoff", ip, port );   // the message to send
      fval=e;
    }
    if(e==1){
      println("valvulaon: "+btn5.getState());
      udp.send("valvulaon", ip, port );   // the message to send
      fval=e;
    }
  }
}

}

void PrimeraConexion(){
if ( msql.connect() )
    {
        msql.query( "SELECT TipoCultivo FROM Cultivo order by TipoCultivo asc" );
        while(msql.next()){
          println( "*** Los Cultivos Son: *** " + msql.getString("TipoCultivo") );
          NombresCultivos += msql.getString("TipoCultivo").trim()+" ";
        }
        println(NombresCultivos);
        list = split(NombresCultivos, ' ');
        printArray(list);
        msql.query( "SELECT count(*) FROM Cultivo" );
        msql.next();
        aux1 = msql.getInt(1);
        println("Registros: "+aux1);
    }
    else
    {
        println("connection failed !");// connection failed !
    }
}
```

Anexos 4. Placa circuital de Arduino Mega 2560



Anexo 5, Sensor de Temperatura DTH11

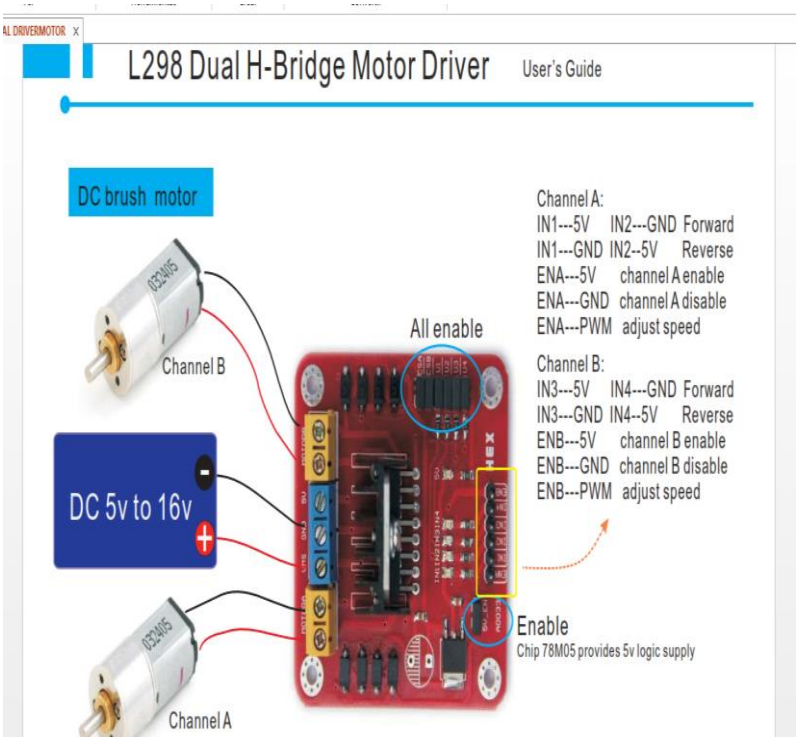


Each DHT11 element is strictly calibrated in the laboratory that is extremely accurate on humidity calibration. The calibration coefficients are stored as programmes in the OTP memory, which are used by the sensor's internal signal detecting process. The single-wire serial interface makes system integration quick and easy. Its small size, low power consumption and up-to-20 meter signal transmission making it the best choice for various applications, including those most demanding ones. The component is 4-pin single row pin package. It is convenient to connect and special packages can be provided according to users' request.

Detailed Specifications:

Parameters	Conditions	Minimum	Typical	Maximum
Humidity				
Resolution		1%RH	1%RH	1%RH
			8 Bit	
Repeatability			±1%RH	
Accuracy	25℃		±4%RH	
	0-50℃			±5%RH
Interchangeability	Fully Interchangeable			
Measurement Range	0℃	30%RH		90%RH
	25℃	20%RH		90%RH
	50℃	20%RH		80%RH
Response Time (Seconds)	1/e(63%)25℃, 1m/s Air	6 S	10 S	15 S
Hysteresis			±1%RH	
Long-Term Stability	Typical		±1%RH/year	
Temperature				
Resolution		1℃	1℃	1℃
		8 Bit	8 Bit	8 Bit
Repeatability			±1℃	
Accuracy		±1℃		±2℃
Measurement Range		0℃		50℃
Response Time (Seconds)	1/e(63%)	6 S		30 S

Anexos 6. L298 Dual



Anexos 7. ENC28J60 Ethernet Controller

MICROCHIP

ENC28J60

Stand-Alone Ethernet Controller with SPI Interface

Ethernet Controller Features

- IEEE 802.3™ Compatible Ethernet Controller
- Fully Compatible with 10/100/1000Base-T Networks
- Integrated MAC and 10Base-T PHY
- Supports One 10Base-T Port with Automatic Polarity Detection and Correction
- Supports Full and Half-Duplex modes
- Programmable Automatic Retransmit on Collision
- Programmable Padding and CRC Generation
- Programmable Automatic Rejection of Erroneous Packets
- SPI Interface with Clock Speeds Up to 20 MHz

Buffer

- 8-Kbyte Transmit/Receive Packet Dual Port SRAM
- Configurable Transmit/Receive Buffer Size
- Hardware Managed Circular Receive FIFO
- Byte-Wide Random and Sequential Access with Auto-Increment
- Internal DMA for Fast Data Movement
- Hardware Assisted Checksum Calculation for Various Network Protocols

Medium Access Controller (MAC) Features

- Supports Unicast, Multicast and Broadcast Packets
- Programmable Receive Packet Filtering and Wake-up Host on Logical AND or OR of the Following:
  - Unicast destination address
  - Multicast address
  - Broadcast address
  - Magic Packet™
  - Group destination addresses as defined by 64-bit Hash Table
  - Programmable Pattern Matching of up to 64 bytes at user-defined offset

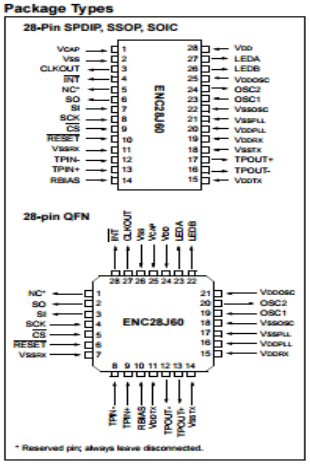
Physical Layer (PHY) Features

- Loopback mode
- Two Programmable LED Outputs for LINK, TX, RX, Collision and Full/Half-Duplex Status

Operational

- Six Interrupt Sources and One Interrupt Output Pin
- 25 MHz Clock Input Requirement
- Clock Out Pin with Programmable Prescaler
- Operating Voltage of 3.1V to 3.6V (3.3V typical)
- 5V Tolerant Inputs
- Temperature Range: -40°C to +85°C Industrial, 0°C to +70°C Commercial (SSOP only)
- 28-Pin SPDIP, SSOP, SOIC, QFN Packages

Package Types



Anexos 8. Formato de encuesta utilizada

UNIVERSIDAD TECNOLÓGICA ISRAEL

CARRERA DE ELECTRÓNICA Y TELECOMUNICACIONES

ENTREVISTA A PROPIETARIOS DE INVERNADEROS

Nombre: Viviana Caiza

Fecha: 01 Febrero 2016

Pregunta 1. ¿Dispone de un invernadero que controle el ambiente interior en forma automática?

SI	
NO	✓

Pregunta 2. Piensa que la tecnología puede facilitar sus actividades diarias en su invernadero, de alguna manera.

SI	✓
NO	

Pregunta 3. ¿Cómo controla la temperatura y humedad de su invernadero?

Manual	✓
Automático	

Pregunta 4. ¿Le gustaría tener en su invernadero un sistema que permita controlar y supervisar la temperatura y humedad de su invernadero de manera automática y remota?

SI	✓
NO	

Pregunta 5. ¿Le parece que este sistema le ahorrara tiempo y controlara el proceso de crecimiento de las plantas de manera óptima?

SI	✓
NO	

Pregunta 6. ¿Invertiría en la implementación del control y supervisión de temperatura y humedad en su invernadero?

SI	✓
NO	