

Invernadero Controlado

Brenda Romero Salcedo

15 de Octubre de 2019

Abstract

El presente documento trata sobre el desarrollo de un “Invernadero Controlado” como parte del curso de Arduino Avanzado 3^a Edición, realizado en la Facultad de Ciencias y organizado por Darwin Eventur, para evaluar los conocimientos adquiridos.

Se ha diseñado un invernadero, de tamaño reducido y con elementos caseros, en el cual se activa o desactiva la función de regadío por goteo de forma manual si se introduce la letra “r”, que activa, o la letra “s”, que desactiva. Además, se han dispuesto otros sensores, tales como de temperatura, de humedad ambiente, de luminosidad y de agua que generan información útil para el control del invernadero. Los datos que generan estos sensores se graficarán una vez obtenidos. La comunicación con el PC se realiza mediante Bluetooth.

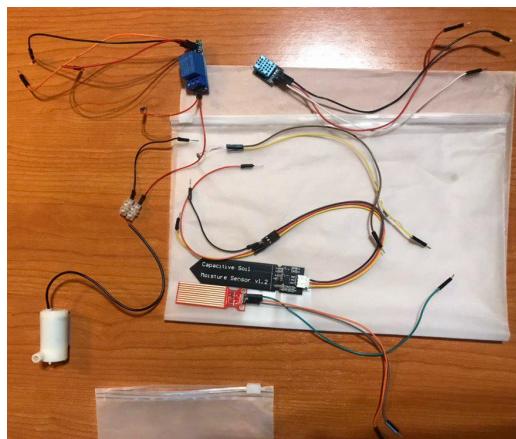


Presentación del invernadero

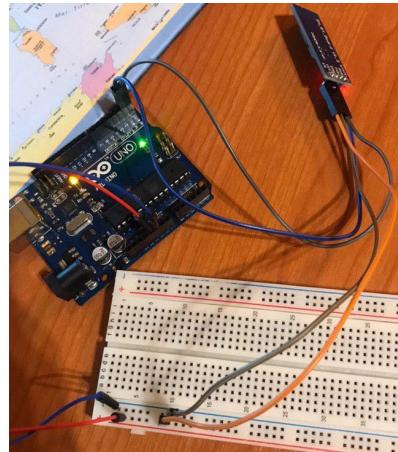
1 Materiales

1.1 Electrónicos

1. Placa de Arduino R3
2. Placa Breadboard
3. Pila 9V
4. Pila adicional de 4,5 V
5. Relé
6. Bluetooth HC-06
7. Sensor Humedad - Temperatura DHT-11
8. Fotorresistencia LDR
9. Resistencia 10kΩ
10. Sensor de agua HOYA
11. Sensor humedad suelo (Capacitive Soil Moisture Sensor v1.2)
12. Bomba de agua junto con goma
13. Relé TONGLING 5VDC
14. Cables macho-macho/hembra macho



Sensores, relé y bomba de agua



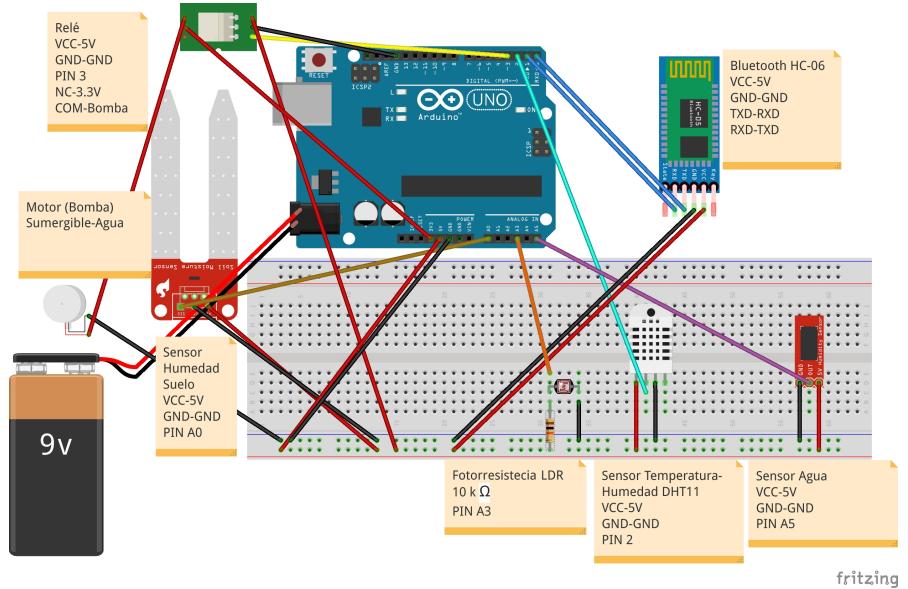
Conexión del dispositivo Bluetooth HC-06

1.2 No electrónicos

1. Caja de plástico de 25L (42x36x25)
2. Fiambrera de plástico (1,5 L)
3. Bandeja de plástico de 4L
4. Palillos de madera
5. Pajitas
6. Cinta aislante
7. Fichas de empalme
8. Caja de cartón pequeña
9. 7 Plantas de distinto tipo
10. Tierra
11. Agua

2 Circuito

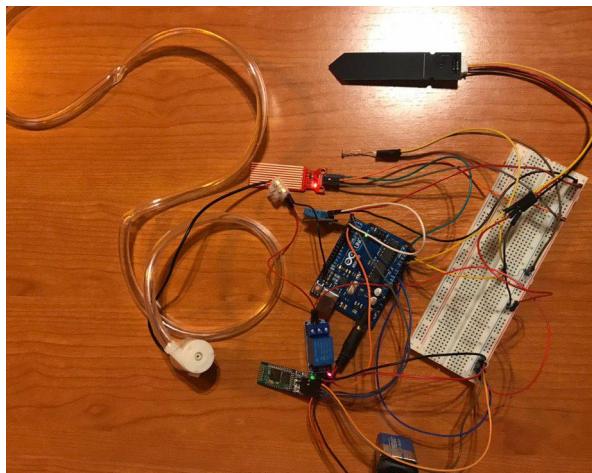
2.1 Dibujo



Dibujo del circuito realizado con Fritzing en el que se puede observar cada uno de los sensores empleados y cómo estos se han conectado al Arduino.

ERRATA: EL BLUETOOTH ESTÁ CONECTADO AL PIN ANALÓGICO A1, A2, no al TX/RX como se hizo inicialmente

2.2 Fotografía



Inicio del montaje del circuito sobre el Breadboard

3 Código

A continuación se muestra detalladamente el código, primero el código principal en el que se observa además de el void setup() y el void loop(), la llamada al fichero “funciones.h”. Segundo se describe “funciones.h” que incluye todas aquellas funciones necesarias para ejecutar el código y controlar la actividad de riego en un invernadero de escala muy reducida.

3.1 Código principal

3.1.1 #include “funciones.h” y void setup() del programa.

```
intemario | Recientes | C:\Users\Jesús\Documents\Arduino\Proyecto\Invernadero\Invernadero.ino
/*
 *Programa que activa el riego de un invernadero por goteo
 * si pulsamos la tecla "r" y lo desactiva si pulsamos la letra "s".Ademas, toma datos de humedad del suelo,
 * de humedad ambiente, de temperatura ambiental,del nivel de agua cubriendo, y de luminosidad.
 * La comunicacion con el PC es Bluetooth al PC.*/
//Libreria con todas las funciones necesarias creadas.
#include "funciones.h"

void setup()
{
    Serial.begin(9600);
    SerialBT.begin(9600);

    dht.setup(2); //Conexion con DHT-11 (sensor temperatura-humedad)
    pinMode(3, OUTPUT); //Conexion con el rele

    //Realizamos una interrupcion por software para que podamos comunicarnos por bluetooth y poder activar/desactivar el riego
    Timer1.initialize(2000000); //Tiempo que dejamos

    Timer1.attachInterrupt(Comunicacion); //Funcion comunicacion

    contador = 0; //Establecimiento de un contador
}
```

En el void setup(), se llama a la función comunicación de manera que nos recoge todos los datos tomados por los distintos sensores, al tiempo, gracias a la interrupción por software, que podemos teclear si queremos regar (“r”) ó no (“s”) en el void loop().

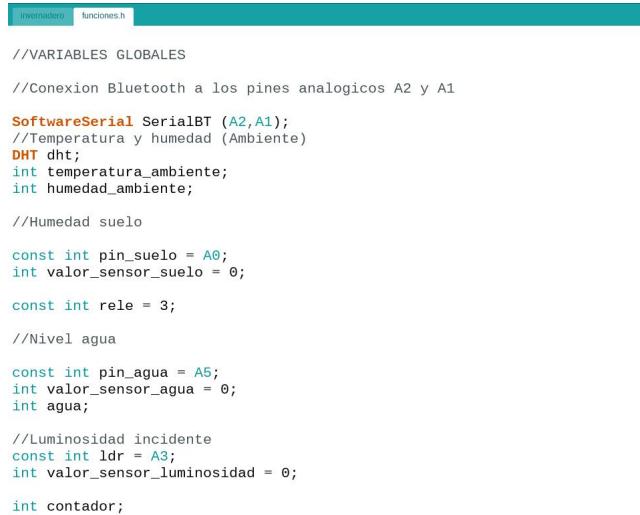
3.1.2 void loop() del programa

```
void loop()
{
    char entrada; //Variable para la letra que vamos a introducir
    //Llamo en cada loop a la funcion temperatura-humedad ya que suele dar problemas con el resto del codigo
    while(temperatura_ambiente == 0) TempHum();
    if (SerialBT.available() > 0) //Funcion para introducir por teclado mediante en la conexion bluetooth
    {
        entrada = SerialBT.read();
        if (entrada == 'r')
        {
            TempHum();
            Regar(); //Funcion regar (Activa el riego cuando introducimos la "r")
        }
        else if (entrada == 's')
        {
            TempHum();
            noRegar(); //Funcion no regar (Desactiva el riego cuando introducimos la "s")
        }
    }
}
```

En el void loop() observamos que nos comunicados por pantalla con el Bluetooth de tal manera que si llamamos a la función Regar(), regamos al introducir “r”, o si por el contrario llamamos a la función noRegar() al haber tecleado la “s”.

3.2 Código de cabecera #include “funciones.h”

3.2.1 Variables globales



```
inviadero | funciones.h

//VARIABLES GLOBALES
//Conexion Bluetooth a los pines analogicos A2 y A1
SoftwareSerial SerialBT (A2,A1);
//Temperatura y humedad (Ambiente)
DHT dht;
int temperatura_ambiente;
int humedad_ambiente;

//Humedad suelo
const int pin_suelo = A0;
int valor_sensor_suelo = 0;
const int rele = 3;

//Nivel agua
const int pin_agua = A5;
int valor_sensor_agua = 0;
int agua;

//Luminosidad incidente
const int ldr = A3;
int valor_sensor_luminosidad = 0;

int contador;
```

En el fichero de cabecera “*funciones.h*” se incluyen todas las variables globales al código y se declaran todas aquellas que son constantes. Al ser globales

serán comunes a todo el código, facilitando la ejecución del mismo, cuando por ejemplo queramos introducir una serie de cambios.

3.2.2 Librerías utilizadas

```
l intermedio funciones.h
/*
 *Fichero de cabecera con las distintas funciones para la toma de datos*/
//LIBRERIAS

//Libreria humedad y temperatura para el sensor dht11
#include <DHT.h>
//Libreria para la interrupcion por software
#include <TimerOne.h>
//Libreria para el control por Bluetooth
#include <SoftwareSerial.h>
```

Las librerías utilizadas han sido: DHT.h para humedad y temperatura, TimerOne.h para la interrupción por software, y SoftwareSerial.h para el Bluetooth.

3.2.3 Funciones de “Riego”

```
l intermedio funciones.h
//FUNCIONES

/*Observese que en cada función llamamos pedimos los datos de temperatura y humedad puesto que suele dar problemas
con una función independiente, por eso se es redundante*/

//Riego activado
void Regar()
{
    digitalWrite(rele, HIGH);

    delay(dht.getMinimumSamplingPeriod());
    temperatura_ambiente = dht.getTemperature();
    humedad_ambiente = dht.getHumidity();

    return;
}

//Riego desactivado
void noRegar()
{
    digitalWrite(rele, LOW);

    delay(dht.getMinimumSamplingPeriod());
    temperatura_ambiente = dht.getTemperature();
    humedad_ambiente = dht.getHumidity();

    return;
}
```

La función de riego, para regar se han escrito dos funciones, una que activa y otra que desactiva; la primera se encarga de mantener en “HIGH” al relé conectado a un pin digital, y la segunda en LOW a dicho relé. El mismo conecta por un lado con el Arduino a un pin digital, en el void setup() se observa que es el pin digital 3, y por el otro lado con una pequeña bomba de agua, al COM(Comunicación) y al NC(Contacto Normalmente Cerrado).

Se han establecido dos funciones independientes para que se llame a una u a otra función para conseguir una mayor sencillez y versatilidad del código.

3.2.4 Funciones de lectura de “Humedad del Suelo” “Temperatura y Humedad Ambiental”

```
//Lectura del sensor de humedad de suelo
int lectura_humedad_suelo()
{
    valor_sensor_suelo = map(analogRead(pin_suelo),0, 1023, 100,0);
    return valor_sensor_suelo;
}

//Lectura del sensor de la temperatura ambiente
int lectura_temperatura_ambiente()
{
    delay(dht.getMinimumSamplingPeriod());
    temperatura_ambiente = dht.getTemperature();
    return temperatura_ambiente;
}

//Lectura del sensor de humedad ambiente
int lectura_humedad_ambiente()
{
    delay(dht.getMinimumSamplingPeriod());
    humedad_ambiente = dht.getHumidity();
    return humedad_ambiente;
}
```

La función de lectura de humedad del suelo, de esta función depende prácticamente toda la ejecución de la acción de riego. Es el parámetro de medida fundamental para determinar si en un invernadero o en cualquier terreno es necesario regar o no. En nuestro caso, a medida que observemos como varían los datos gráficamente, decidiremos si regar o no. Así, esta función mide el valor del sensor conectado a un pin analógico (A0), cuyo valor se ha mapeado de 0 a 100 para considerarlo como porcentaje.

Por otra parte, para medir la temperatura y la humedad ambiental se utiliza el sensor DHT-11, que se conecta al Arduino al pin digital 2. Se han escrito dos funciones independientes a partir del mismo sensor con el fin de usar ambas medidas de modo independiente. Cada función devuelve el valor de la temperatura y de la humedad ambiente.

3.2.5 Funciones de lectura de la “Luminosidad y de Agua”

```
//Lectura del sensor de luminosidad
int lectura_luminosidad()
{
    valor_sensor_luminosidad = map(analogRead(ldr), 0,1023, 100,0);
    int temperatura = lectura_temperatura_ambiente();
    return valor_sensor_luminosidad;
}

//Lectura del sensor de agua
int lectura_nivel_agua()
{
    valor_sensor_agua = analogRead(pin_agua);
    agua=valor_sensor_agua*100/1023;
    return (agua);
}
```

Se ha creado un función dedicada a la medida de la luminosidad incidente utilizando un fotorreceptor LDR conectado al pin analógico A3, el valor de

luminosidad se mapea de 0 a 100 para observar el porcentaje de luminosidad que llega al invernadero. Conocer qué cantidad de luz recibimos es muy útil para relacionarlo con la temperatura.

No obstante, la medida del nivel del agua, sirve para controlar el agua en el recipiente como resultado de regar las plantas. El sensor para la misma se conecta al pin analógico A5, valor que se pasa de 0 a 100 en lugar de 0 a 1023, obteniendo un porcentaje que al ser analizado gráficamente podremos prever una posible inundación y parar el riego.

3.2.6 Función de comunicación por Bluetooth

```
//Comunicacion Bluetooth-PC
void Comunicacion ()
{
    lectura_humedad_suelo();
    lectura_nivel_agua();
    lectura_luminosidad();
    lectura_humedad_ambiente();
    lectura_temperatura_ambiente();
    lectura_humedad_suelo();

    SerialBT.print(contador); SerialBT.print("\t"); SerialBT.print(valor_sensor_suelo); SerialBT.print("\t"); SerialBT.print(temperatura_ambiente); SerialBT.print("\t");
    SerialBT.print(humedad_ambiente);SerialBT.print("\t"); SerialBT.print(valor_sensor_luminosidad); SerialBT.print("\t"); SerialBT.print(agua);
    SerialBT.println();
}

contador++;

return;
}

void TempHum ()
{
    delay(dht.getMinimumSamplingPeriod());
    temperatura_ambiente = dht.getTemperature();
    humedad_ambiente = dht.getHumidity();

    return;
}
```

Todos los datos que toman los sensores se comunican al PC a través de Bluetooth. En mi caso, se utiliza modelo HC-06. EL cual he conectado el RX y TX a los pines analógicos A2, A1. Además de conectarlo al VCC y al GND pertinentes. Inicialmente los conecté al RX y al TX pero me generaba bastantes problemas de comunicación.

La función `comunicacion()`, función que cuando es evocada por el código principal, la cual tiene como argumentos todo aquello que es devuelto por todas las funciones descritas anteriormente más el valor de un contador que corresponde con el `void loop()` del código principal, imprime por pantalla en columnas todos los valores medidos, que servirán para informar y para graficarse. Por último, la función `TempHum()` no es más que una función redundante que toma de nuevo los valores de temperatura y de humedad ambientales.

4 Comunicación

La comunicación de los datos, como ya se ha mencionado, se realiza por Bluetooth.

4.1 Bluetooth

Para configurar el Bluetooth en mi caso, ejecuto directamente sobre un terminal Linux lo siguiente:

- **sudo service bluetooth restart**
- **sudo rfcomm connect 0 00:14:03:00:02:27** (MAC de mi dispositivo)

rfcomm nos permite conectar con el dispositivo, 0 es el canal (podría ser otro), y los restantes números corresponden con el MAC del dispositivo, dicha MAC la podemos obtener introduciendo el comando hcitool scan.

4.2 Graficar los datos recibidos

Se ha editado un script de shell, antes de ejecutar cualquier script es necesario dar permisos con el comando chmod y ser sudo, si no, no funcionará, *invernadero_PLOT.sh*. Este script genera una serie de gráficas que nos permite observar y controlar gráficamente el invernadero ya que nos permite ver cuales son las necesidades hidráticas de las plantas en función de los valores de los distintos parámetros que hemos medido.

Para graficar desde el propio terminal de Linux, y habiendo establecido previamente la comunicación por Bluetooth, nos localizarnos en el mismo directorio donde tengamos todos los ficheros necesarios para poder graficar e introducimos lo siguiente para nuestro caso:

- **cat /dev/rfcomm0 >> invernadero_DATOS.lis & ./invernadero_PLOT.sh**

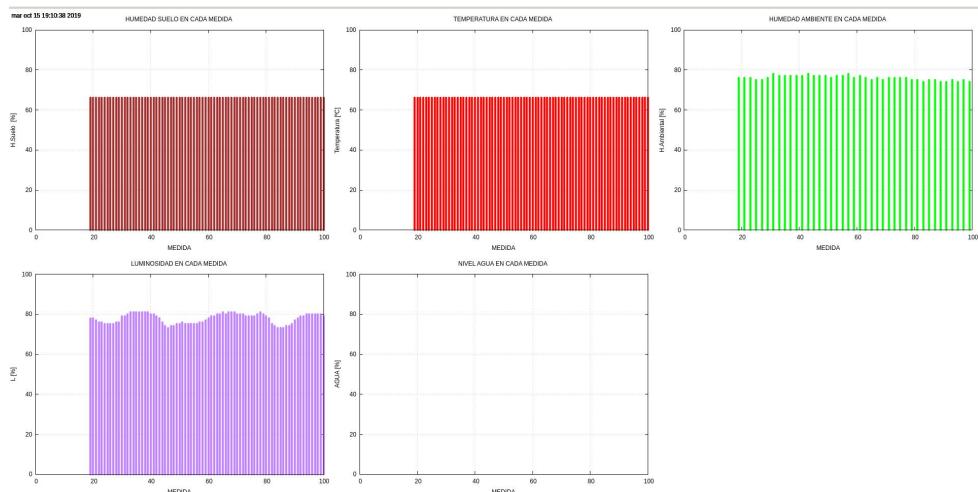


Imagen resultante de graficar utilizando GNUPLOT, se puede observar cada uno de los distintos sensores dispuestos en el invernadero, correspondientes a las 5 gráficas. Asimismo, es muy importante ver que en el último gráfico no se gráfica nada puesto que el nivel de agua en el recipiente era de 0 y no ha habido ninguna fuga. Por otro lado, los primeros 20 datos no se grafican bien puesto que se producen ciertos errores cuando llamamos a graficar, probablemente por el inicio de establecimiento de la comunicación.

5 EL INVERNADERO

5.1 Proceso de montaje



Instalación de la bomba de agua en el recipiente de plástico



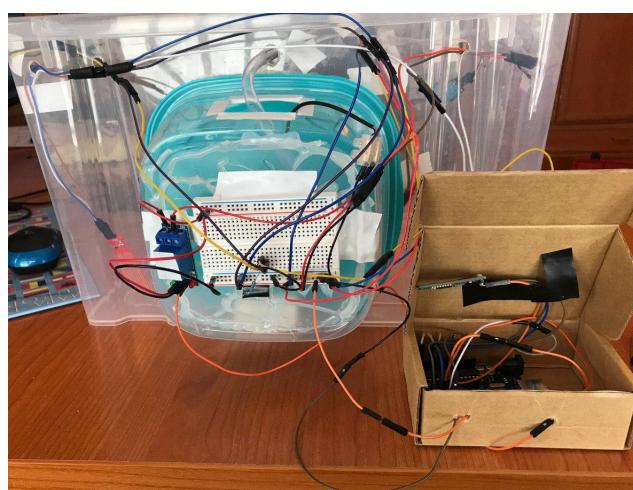
Plantas transplantadas y realización de agujeros “desagüe”



Disposición de las plantas que van a ser introducidas en el invernadero



Bomba instalada junto con el breadboard, y demás sensores



Bomba e instalación del Arduino en una caja como protección



Vista lateral de la instalación: obsérvese las plantas y los sensores en interior conectados



Interior del invernadero, señalar que la goma de agua ha sido agujereada para crear el riego por goteo, el extremo de la goma ha sido tamponado



6 Conclusiones

6.0.1 Sugerencias

La sugerencia principal para este proyecto, desde mi punto de vista, sería el añadir un sensor de CO₂, pero el sensor MG811, diseñado para ello, tiene un coste bastante elevado. No obstante, es posible utilizar el sensor MQ135 que mide gases de polución. Sin embargo, no estaba totalmente segura si era muy eficiente en relación al CO₂ ya que es mejor para medir CO y NO_x. Es por ello que no lo he aplicado.

6.0.2 Conflictos

Se trató de conectar una pantalla LCD pero su conexión provocaba interferencia con el resto de sensores y dispositivos conectados a la placa de Arduino, siendo además necesaria un mayor suministro extra de energía. Asimismo, la librería `<LiquidCrystal_I2C>`, generaba bastantes problemas con el resto del código, principalmente con la librería para `<dht.h>`. La idea de añadir una pantalla, era el de informar sobre la temperatura ambiente y la humedad del suelo en el invernadero, como método de control visual, lo que además, no lo hace imprescindible.

El sensor DHT-11 da numerosos problemas interfiriendo en el funcionamiento del resto de sensores y del propio código. Es por ello que se ha sido muy redundante con este sensor en cada una de las funciones.

En relación a la configuración bluetooth, ha sido bastante complicado, en un inicio realicé una comunicación por el TX y el RX, pero obtuve por usar la librería `<SoftwareSerial.h>` y conectarlo a los pines analógicos A2, A1.

6.0.3 Código que se adjunta

- *funciones.h*
- *invernadero.ino*
- *invernadero_PLOT.sh*
- “Video del proyecto en funcionamiento”