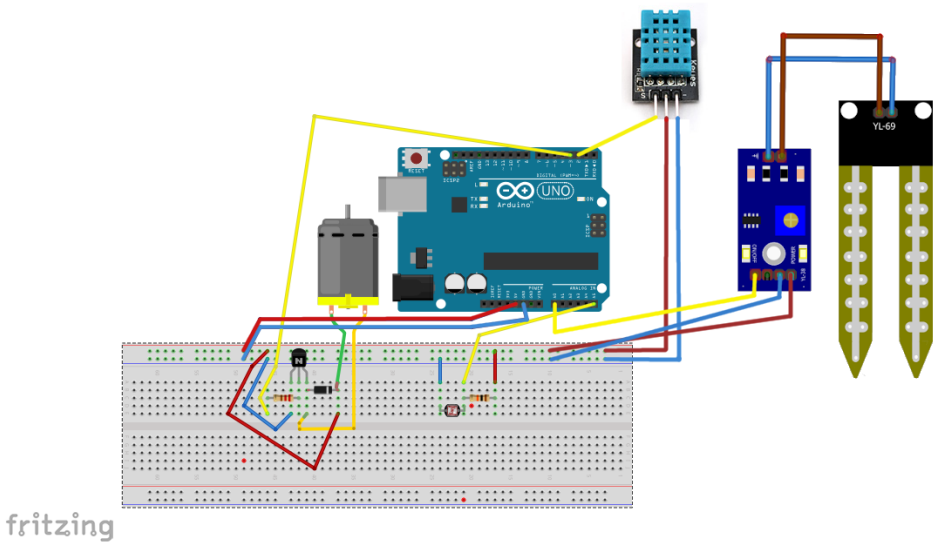
TRABAJO DE INFORMÁTICA

Integrantes del equipo

- Daniel Sánchez García – danielsanchezg.
- Sara Rodríguez Fernández – sararodriguezfernandez.
- Marcos Lozano Rodríguez – marcoslozanorodriguez.

Objetivo del trabajo, sistema de riego.

Nuestro proyecto se basa en crear un sistema de riego automático que trabaje en función de la humedad del suelo y la temperatura ambiente, en función de unos valores establecidos en arduino, de modo que si el sistema se ha puesto en funcionamiento el ordenador mostrará la hora, la fecha, y los valores de temperatura y humedad.



Funcionamiento

La aplicación dispone de un sensor encargado de leer la temperatura y la humedad, mediante el sensor de humedad y temperatura recogemos los valores de temperatura y humedad, de tal forma que si la temperatura está entre 20 y 30 grados Celsius y el porcentaje de humedad se encuentra entre el 20% y el 40%, el riego se activará durante un tiempo, en este caso para simular el riego se utilizará un LED. El ordenador recibe

estos datos y muestra la hora, la fecha y los valores registrados cuando se ha activado el riego.

Hardware

El sensor que ha sido utilizado en el proyecto es el siguiente:

- Sensor de temperatura y humedad relativa en el aire DHT22.

El DHT22 es un sensor, que permite realizar la **medición simultánea de temperatura y humedad**.

Estos sensores disponen de un procesador interno que realiza el proceso de medición, proporcionando la medición mediante una señal digital, por lo que **resulta muy sencillo obtener la medición desde un microprocesador como Arduino**. Las características más importantes de estos sensores son:

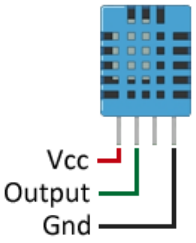
Medición de temperatura entre -40 a 125, con una precisión de 0.5ºC

Medición de humedad entre 0 a 100%, con precisión del 2-5%.

Frecuencia de muestreo de 2 muestras por segundo (2 Hz)

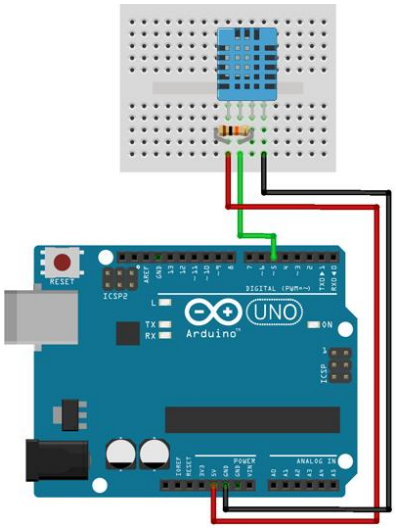
EL DHT22 (sin llegar a ser en absoluto un sensor de alta precisión) **tiene unas características aceptables** para que sea posible emplearlo en proyectos reales de monitorización o registro, que requieran una precisión media.

Estos sensores disponen de 4 patillas, de las cuales usaremos 3, Vcc, Output y GND.

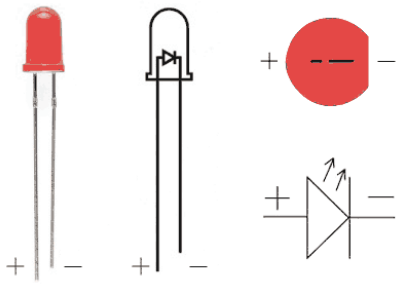


Conectar el sensor es sencillo, simplemente alimentamos desde Arduino al sensor a través de los pines GND y Vcc del mismo. Por otro lado, conectamos la salida Output a una entrada digital de Arduino. Necesitaremos poner una resistencia de 10K entre Vcc y el Pin Output.

El esquema en una protoboard sería:



- LED



Un LED es un diodo emisor de luz, es decir, **un tipo particular de diodo que emite luz** al ser atravesado por una corriente eléctrica. Los diodos (emisor de luz, o no) son unos de los dispositivos electrónicos fundamentales.

Recodemos que **diferenciamos entre dispositivos eléctricos y electrónicos**. Los dispositivos eléctricos engloban resistencias, condensadores y bobinas, e integran el campo de electricidad. Los dispositivos electrónicos, surgen del uso de materiales semiconductores, y dan lugar al campo de la electrónica.

Un diodo es una **unión de dos materiales semiconductores con dopados distintos**. Sin entrar en detalles, esta diferencia de dopado hace que genere una barrera de potencial, que como primera consecuencia hace que el paso de corriente en uno de los sentidos no sea posible.

Aquí tenemos la primera característica de los diodos, tienen polaridad, es decir, **solo dejan pasar la corriente en un sentido**. Por tanto, tenemos que conectar correctamente la tensión al dispositivo.

La patilla larga debe ser conectada al voltaje positivo (ánodo), y la corta al voltaje negativo (cátodo).

Lo principal para hacer funcionar un LED **es calcular el valor de la resistencia necesaria**. Para calcular el valor de tensión necesaria para alimentar un LED necesitamos conectar 3 parámetros

- La tensión de alimentación (Vcc)
- La tensión de polarización directa del LED (Vd)
- La corriente nominal del LED (In)

Calcula el valor de la resistencia es sencillo. Como hemos dicho, la tensión que soporta el LED es la diferencia entre la tensión aplicada y la tensión de polarización directa del LED.

Aplicando la ley de Ohm, con el valor de la intensidad nominal del LED

$$V = V_{cc} - V_d = I_{nominal} * R$$

Por lo que lo que el valor de la resistencia resulta

$$R = \frac{V_{cc} - V_d}{I_{nominal}}$$

Dado que las resistencias comerciales tienen valores normalizados, no encontraréis una resistencia con el valor exacto que hayáis calculado. En este caso, **elegiremos la**

resistencia normalizada inmediatamente superior al valor calculado, para garantizar que la corriente es inferior a la nominal.

Dado que las salidas digitales de Arduino proporcionan una tensión de 5V, la resistencia limitadora de corriente para 20mA sería de 150Ω. El montaje con la protoboard sería el siguiente:

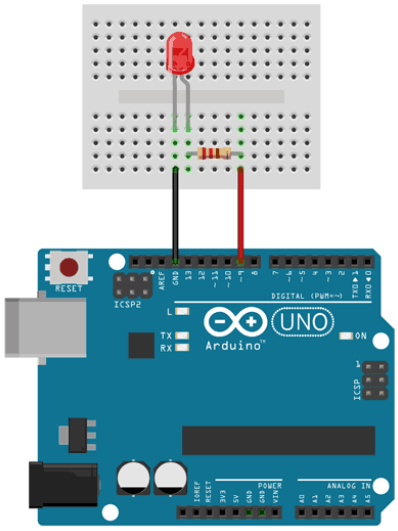
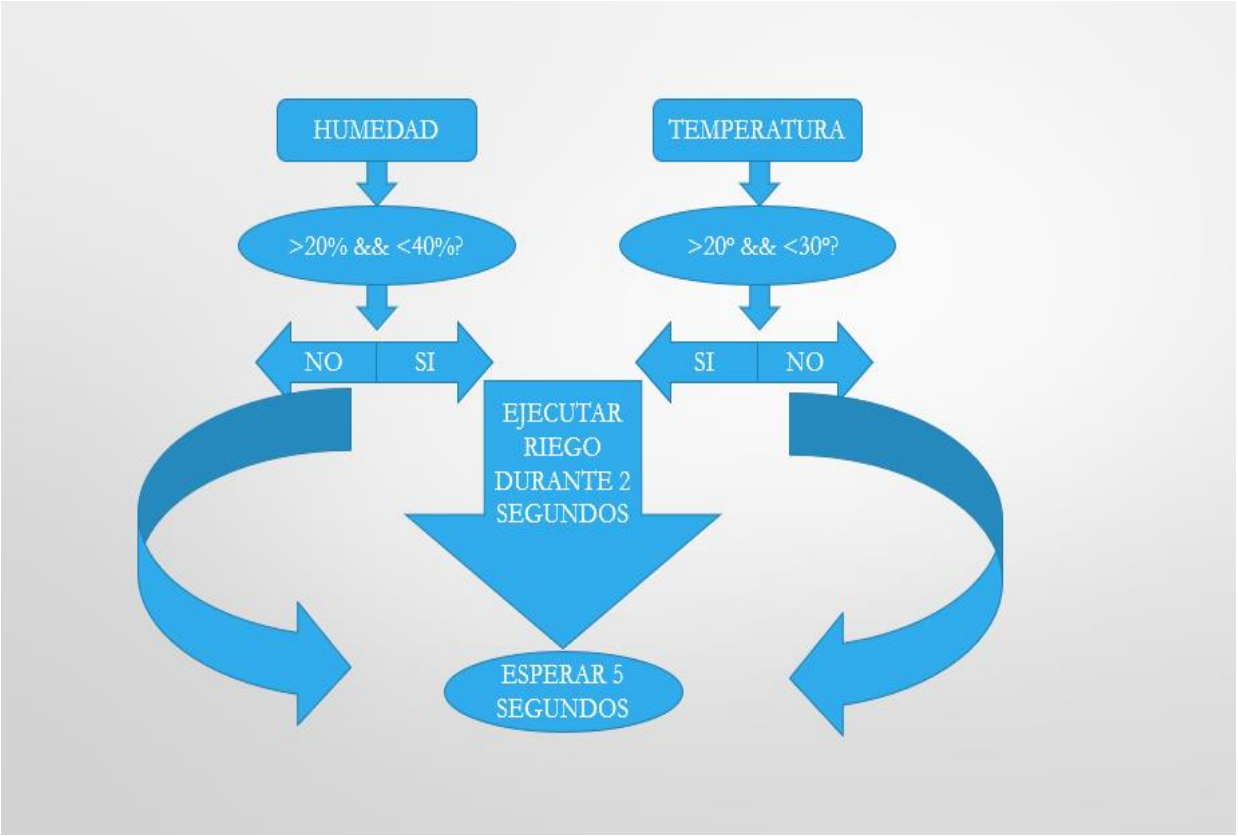


Diagrama de la aplicación

En el siguiente esquema se muestra el bucle de funcionamiento del sensor que capta la temperatura y la humedad.



Funciones utilizadas

Las funciones que han sido usadas son las siguientes:

- Función para devolver la hora: struct tm* time(time_t) .
- Función para la fecha: size_t strftime(char *,size_t,char *,struct tm *).
- Función para la conexión ordenador-arduino: void autoConnect(SerialPort arduino,char).

Comunicación Arduino-DevC

Código en Arduino

Incluimos la biblioteca para el sensor DHT22 y definimos las salidas digitales donde hemos conectado el LED (en la 8) y el DHT22 (en la 2) y los parámetros máximos de temperatura y humedad.

```
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT22

const int ledPIN = 8;
const float maxh = 40.00;
const float minh = 20.00;
const float maxt = 30.00;
const float mint = 20.00;

DHT dht(DHTPIN, DHTTYPE);

void setup()
{
  Serial.begin(9600);//iniciar puerto serie

  pinMode(ledPIN , OUTPUT);  //definir pin como salida
  dht.begin();

}

void loop()
{

  // Leer porcentaje de humedad
  float h = dht.readHumidity();
  // Leer temperatura en Celsius
  float t = dht.readTemperature();
  // Esperar un minuto entre medidas

  if (isnan(h) || isnan(t))
  {
    Serial.println(F("Error en la lectura del sensor"));
    return;
  }


  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Humedad: "));
  Serial.print(h);
  Serial.print(F("%  Temperatura: "));
  Serial.print(t);
  Serial.print(F(" Grados C"));

  Serial.print(hic);
  Serial.println(F(" Grados C "));

  if (t < maxt && t > mint && h < maxh && h > minh)
  {
  digitalWrite(ledPIN , HIGH);    // poner el Pin en HIGH
  delay(2000);                    // esperar un segundo
  digitalWrite(ledPIN , LOW);     // poner el Pin en LOW
  delay(1000); // esperar un segundo
  }
// Tiempo entre medidas
  delay(5000);
}
```

Código en Dev c++

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include "SerialPort.h"
#define MAX_DATA_LENGTH 255
// Funciones prototipo
void autoConnect(SerialPort *arduino,char*);
int main(void)

{

 //Arduino SerialPort object
 SerialPort *arduino;
 // Puerto serie en el que está Arduino
 char* portName = "\\\\.\\COM7";
 // Buffer para datos procedentes de Arduino
 char incomingData[MAX_DATA_LENGTH];

 // Crear estructura de datos del puerto serie
 arduino = (SerialPort *)malloc(sizeof(SerialPort));
 // Apertura del puerto serie
 Crear_Conexion(arduino,portName);
 autoConnect(arduino,incomingData);
}
void autoConnect(SerialPort *arduino,char *incomingData)
{
 int readResult;
  time_t t;
  struct tm *tm;
  char fecha[100];

// Espera la conexión con Arduino
while (!isConnected(arduino))
{
Sleep(100);
Crear_Conexion(arduino,arduino->portName);
}
 //Comprueba si arduino está connectado
if (isConnected(arduino))
{
printf ("Conectado con Arduino en el puerto %s\n",arduino->portName);
printf ("\n");
}
 // Bucle de la aplicación
while (isConnected(arduino))
{
 readResult = readSerialPort(arduino,incomingData, MAX_DATA_LENGTH);
if (readResult!=0)
{
  t=time(NULL);
  tm=localtime(&t);
  strftime(fecha, 100, "%d/%m/%Y", tm);
  printf ("El sistema ejecuto el riego el %s ", fecha);
  printf ("a las %d:%d:%d\n",tm->tm_hour,tm->tm_min,tm->tm_sec);
  printf("%s \n", incomingData,readResult);
 Sleep(10);
}
}
if (!isConnected(arduino))
 printf ("Se ha perdido la conexion con Arduino\n");

}
```

Fuentes:

https://www.luisllamas.es/arduino-dht11-dht22/

https://naylampmechatronics.com/sensores-temperatura-y-humedad/58-sensor-de-temperatura-y-humedad-relativa-dht22-am2302.html

https://www.luisllamas.es/encender-un-led-con-arduino/

Archivos de la asignatura colgados en moodle