FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS  
HUGO VEGA HUERTA

INTELIGENCIA ARTIFICIAL

**Sistema Inteligente de Monitoreo y Control para un Huerto utilizando Sensores de Humedad, Temperatura y Lluvia**

Giuseppe Jefferson Cordova Silva, Ivan Paolo De La Cruz Guillen, Manuel Hernan Gonzales Rojas, Jose Luis Perez Grados, Hector Imanol Rafael Javier, Karen Antonia Rojas Hurtado, Alejandro Paul Torres Espinoza, Jamie Edinso Vilchez Giraldo

Universidad Nacional Mayor de San Marcos

Facultad de Ingeniería de Sistemas e Informatica

giuseppe.cordova@unmsm.edu.pe, ivan.delacruz1@unmsm.edu.pe, manuel.gonzales13@unmsm.edu.pe, jose.perez45@unmsm.edu.pe, hector.rafael@unmsm.edu.pe, karenantonia.rojas@unmsm.edu.pe, alejandro.torres4@unmsm.edu.pe, jamie.vilchez@unmsm.edu.pe

# **RESUMEN**

En este artículo se presenta el desarrollo de un sistema inteligente de huerto que utiliza sensores de humedad, temperatura y lluvia en combinación con una placa Arduino UNO. Estos sensores permiten el monitoreo en tiempo real de las condiciones ambientales, lo que facilita la toma de decisiones informadas para la gestión agrícola. La programación se realiza en el entorno de desarrollo Arduino IDE, donde se configura la placa Arduino UNO para recopilar y procesar datos de los sensores. Esta integración de tecnología y agricultura busca transformar un huerto convencional en un sistema automatizado y eficiente, explorando el potencial de la tecnología en la agricultura moderna.

**Palabras clave:** Sensor de humedad, sensor de temperatura, sensor de lluvia, Arduino UNO, huerto, programación, tecnología, agricultura.

# **ABSTRACT**

This paper presents the development of a smart orchard system using humidity, temperature and rainfall sensors in combination with an Arduino Uno board. These sensors allow real-time monitoring of environmental conditions, which facilitates informed decision making for agricultural management. Programming is done in the Arduino IDE development environment, where the Arduino Uno board is configured to collect and process data from the sensors. This integration of technology and agriculture seeks to transform a conventional orchard into an automated and efficient system, exploring the potential of technology in modern agriculture.

**Key words:** Humidity sensor, temperature sensor, rain sensor, Arduino UNO, orchard, programming, technology, agriculture.

1. **Introducción**

En este trabajo, presentamos el desarrollo de un huerto inteligente que representa la convergencia de la tecnología y la agricultura. Utilizando una placa Arduino Uno en combinación con sensores de temperatura, humedad y lluvia, de esta manera exploraremos cómo la tecnología puede transformar un huerto convencional en un sistema automatizado y eficiente.

Los sensores desempeñan un papel central en la concepción de nuestro proyecto. El sensor de temperatura nos permite monitorear las condiciones térmicas en tiempo real, mientras que el sensor de humedad nos proporciona información esencial sobre el contenido de humedad en el suelo. Por último, el sensor de lluvia detecta la precipitación. Estos sensores trabajan en conjunto con Arduino, que asume el papel de cerebro de nuestro huerto inteligente.

La programación de Arduino se realiza en el Arduino IDE, un entorno de desarrollo integrado que se convierte en nuestro taller digital. A través del Arduino IDE, configuramos la placa Arduino Uno para recopilar y procesar los datos provenientes de los sensores de temperatura, humedad y lluvia. Además, en este entorno, damos forma a los algoritmos de inteligencia artificial que capacitan a nuestro sistema para tomar decisiones informadas en tiempo real. Esta sinergia entre la electrónica, la programación y la agricultura sienta las bases para un proyecto innovador y prometedor que explora las posibilidades de la tecnología en el ámbito agrícola.

Para el desarrollo de nuestro trabajo hemos utilizado las siguientes materiales:

* Sensor de temperatura
* Sensor de humedad dht11
* Sensor de lluvia
* Motor con controlador 28byj-48
* Leds
* Arduino Uno
* Protoboard
* Jumpers

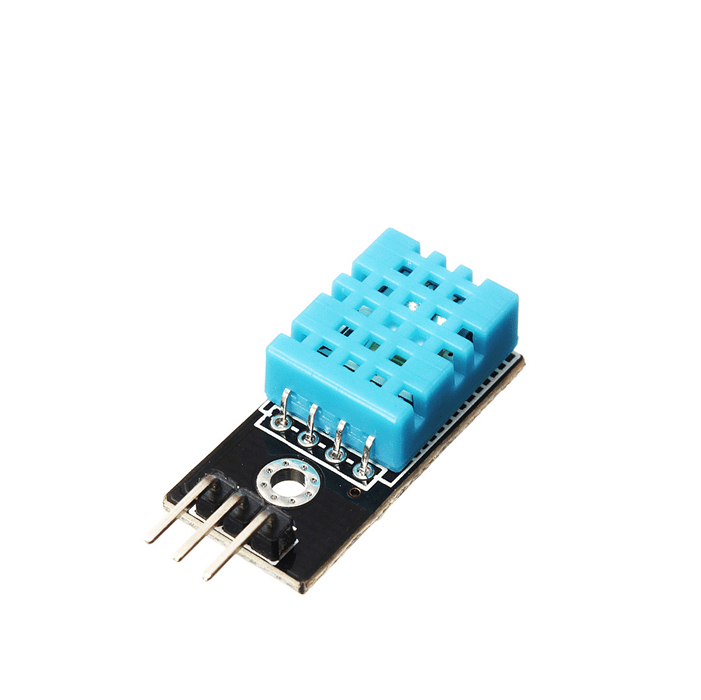
1. **Fundamentación Teórica**

**2.1. Definición de Sensor de Temperatura**

Un sensor de temperatura compatible con Arduino es un dispositivo electrónico diseñado para captar la temperatura del ambiente o de un objeto en particular. Este componente transforma las alteraciones térmicas en señales eléctricas utilizables para supervisar y regular la temperatura en proyectos y sistemas electrónicos.

Estos sensores de temperatura emplean una variedad de mecanismos para detectar los cambios en la temperatura. Entre ellos, el termistor es uno de los más comunes, caracterizado por modificar su resistencia eléctrica en respuesta a las variaciones térmicas. Otros tipos abarcan sensores de temperatura de estado sólido, sensores infrarrojos y sensores de temperatura digital.

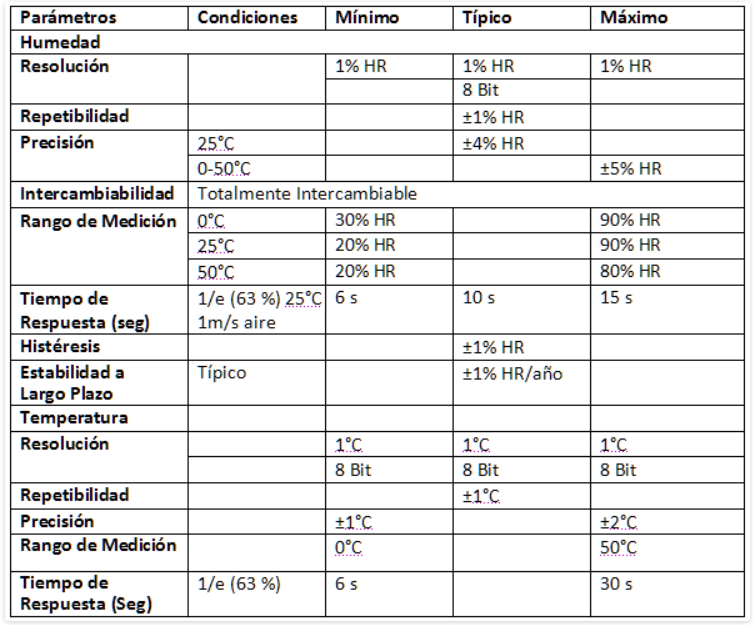
La exactitud de un sensor de temperatura puede variar dependiendo de su tipo y modelo específico. Mientras algunos sensores brindan mediciones altamente precisas que son ideales para aplicaciones que demandan una precisión extrema, otros son más adecuados para usos generales que no requieren una precisión extrema. Estos se vinculan con la placa Arduino a través de sus pines de entrada y salida. Los sensores digitales, en su mayoría, comunican la temperatura en forma de un valor numérico que Arduino puede leer directamente. En cambio, los sensores analógicos producen una señal eléctrica proporcional a la temperatura que necesita ser convertida en un valor numérico mediante la lectura del puerto analógico de Arduino.



**Figura N° 1.** Sensor de DHT11

**2.2. Definición de Sensor de Humedad dht11**

El sensor de humedad DHT11 posee una capacidad para medir la humedad relativa y la temperatura ambiente de manera precisa y accesible, este pequeño componente se ha convertido en un elemento fundamental en proyectos de Arduino y otros microcontroladores.



**Figura N° 2.** Datos de sensor de temperatura

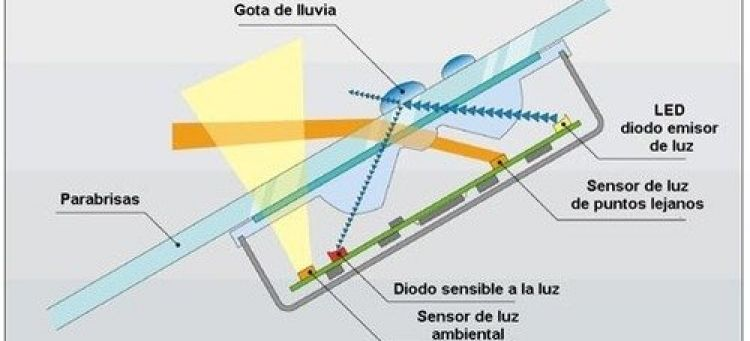
El DHT11 opera gracias a un ingenioso principio de funcionamiento. Para medir la temperatura, hace uso de un termistor, un componente resistivo cuya resistencia eléctrica varía con la temperatura. Los cambios en la resistencia se traducen en valores numéricos que reflejan la temperatura ambiente. Por otro lado, la medición de humedad se basa en un elemento resistivo sensible a la humedad. A medida que la humedad fluctúa, esta característica se refleja en la resistencia eléctrica, proporcionando datos precisos sobre la humedad relativa del aire.

El DHT11 exhibe características imprescindibles para proyectos electrónicos:

* Amplio Rango de Medición
* Versatilidad Térmica
* Precisión Apreciada
* Interfaz Sencilla
* Aplicaciones
* Control Ambiental Residencial
* Jardinería y Agricultura
* Meteorología Casera
* Automatización

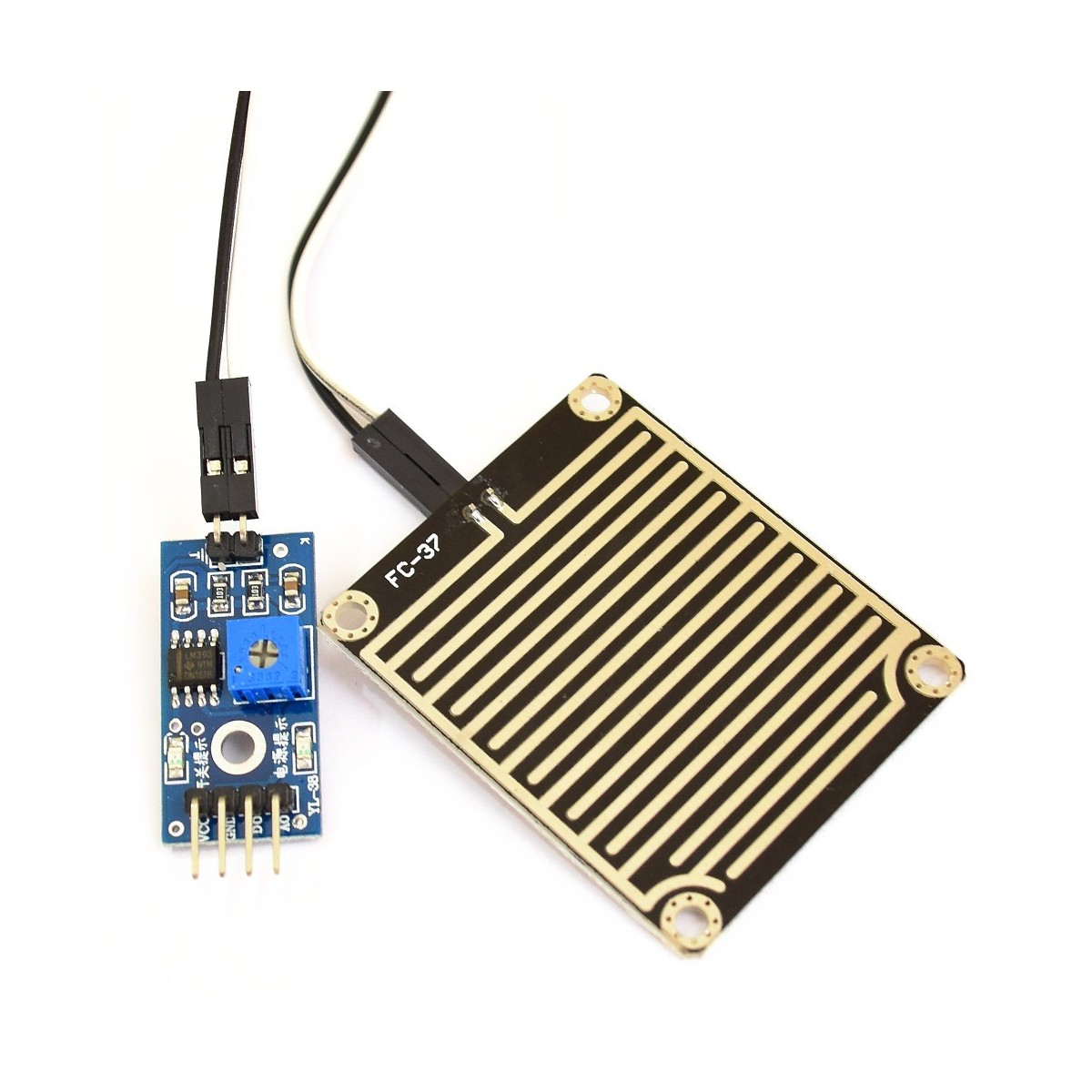
**2.3. Sensor de Lluvia**

El sensor de lluvia de Arduino es un componente electrónico especializado diseñado para detectar y cuantificar la presencia de precipitación, específicamente la lluvia. Este dispositivo es altamente apreciado en proyectos de automatización, meteorología y control de sistemas de riego, ya que brinda la capacidad de convertir la lluvia en datos eléctricos procesables.



**Figura N° 3.** Funcionamiento del Sensor de lluvia

El funcionamiento de este sensor se basa en una tecnología sensible a la humedad que se encuentra en su superficie. Cuando la lluvia cae sobre el sensor, el agua actúa como un conductor eléctrico y altera la resistencia eléctrica en la superficie del sensor. Esta modificación de la resistencia se traduce en una señal eléctrica proporcional a la cantidad de lluvia que está cayendo en ese momento. Cuanto mayor sea la cantidad de agua que caiga sobre el sensor, menor será la resistencia eléctrica y viceversa.

**Figura N° 4.** Sensor de Lluvia

Características Clave:

Sensibilidad a la lluvia: Los sensores de lluvia de Arduino son altamente sensibles y pueden detectar incluso lluvias ligeras y lloviznas.

Salida Analógica o Digital: Dependiendo del modelo, estos sensores pueden proporcionar una salida analógica que varía en función de la cantidad de lluvia o una salida digital que indica simplemente si está lloviendo o no.

Compatibilidad con Arduino: Estos sensores se pueden conectar de manera sencilla a las placas Arduino, y los datos de lluvia detectada se pueden leer y procesar con facilidad.

Versatilidad de Aplicación: Se utilizan en una variedad de aplicaciones, desde sistemas de alarma de lluvia hasta sistemas de riego inteligentes y estaciones meteorológicas caseras.

Aplicaciones Comunes:

* Sistemas de Riego Automatizado
* Control de Ventanas y Techos
* Alertas Meteorológicas
* Control de Iluminación
* Proyectos de Monitoreo Ambiental

**2.4. Definición de Motor con controlador 28byj-48**

El motor con controlador 28BYJ-48 es un componente fundamental en el mundo de la automatización y la robótica. Este conjunto se compone de un motor paso a paso (28BYJ-48) y un controlador que permite una precisión y un control excepcionales sobre el movimiento mecánico. Ampliamente utilizado en proyectos de electrónica y robótica, este motor con controlador se destaca por su capacidad para realizar movimientos de avance y retroceso con un alto grado de exactitud y control, lo que lo convierte en un recurso invaluable para una variedad de aplicaciones.

Principio de Funcionamiento:

El motor paso a paso 28BYJ-48 funciona mediante la activación secuencial de bobinas electromagnéticas en su interior. Cada paso del motor corresponde a una rotación predeterminada y precisa, lo que permite un control fino del ángulo y la velocidad del giro. El controlador se encarga de coordinar estos movimientos, lo que garantiza una operación fluida y controlada del motor.

Características Clave:

Alta Precisión: Este motor es conocido por su alta precisión y la capacidad de moverse con incrementos definidos, lo que facilita tareas que requieren control detallado del movimiento.

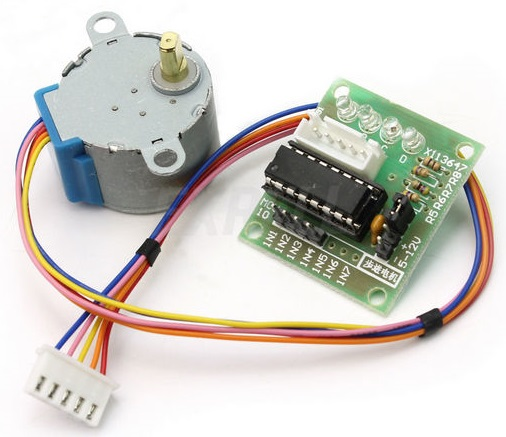
Control por Pasos: El motor con controlador 28BYJ-48 permite avanzar o retroceder en pasos definidos, lo que simplifica la programación y control del movimiento.

Amplia Aplicabilidad: Se utiliza en una amplia variedad de aplicaciones, desde la automatización de sistemas de cámara y enfoque automático hasta la construcción de brazos robóticos y proyectos de posicionamiento.

Compatibilidad Electrónica: Es fácilmente compatible con placas Arduino y otros microcontroladores, lo que simplifica su integración en proyectos electrónicos.

Aplicaciones Comunes:

* Impresoras 3D
* Robótica
* Cámaras y Fotografía
* Proyectos de Automatización



**Figura N° 5.** Motor con controlador 28byj-48

**2.5. Definición de Leds**

Un LED, o diodo emisor de luz, es un dispositivo semiconductor que emite luz cuando se aplica una corriente eléctrica. A diferencia de las bombillas incandescentes tradicionales, los LEDs no requieren filamentos y son altamente eficientes en la conversión de energía eléctrica en luz.

Principio de Funcionamiento:

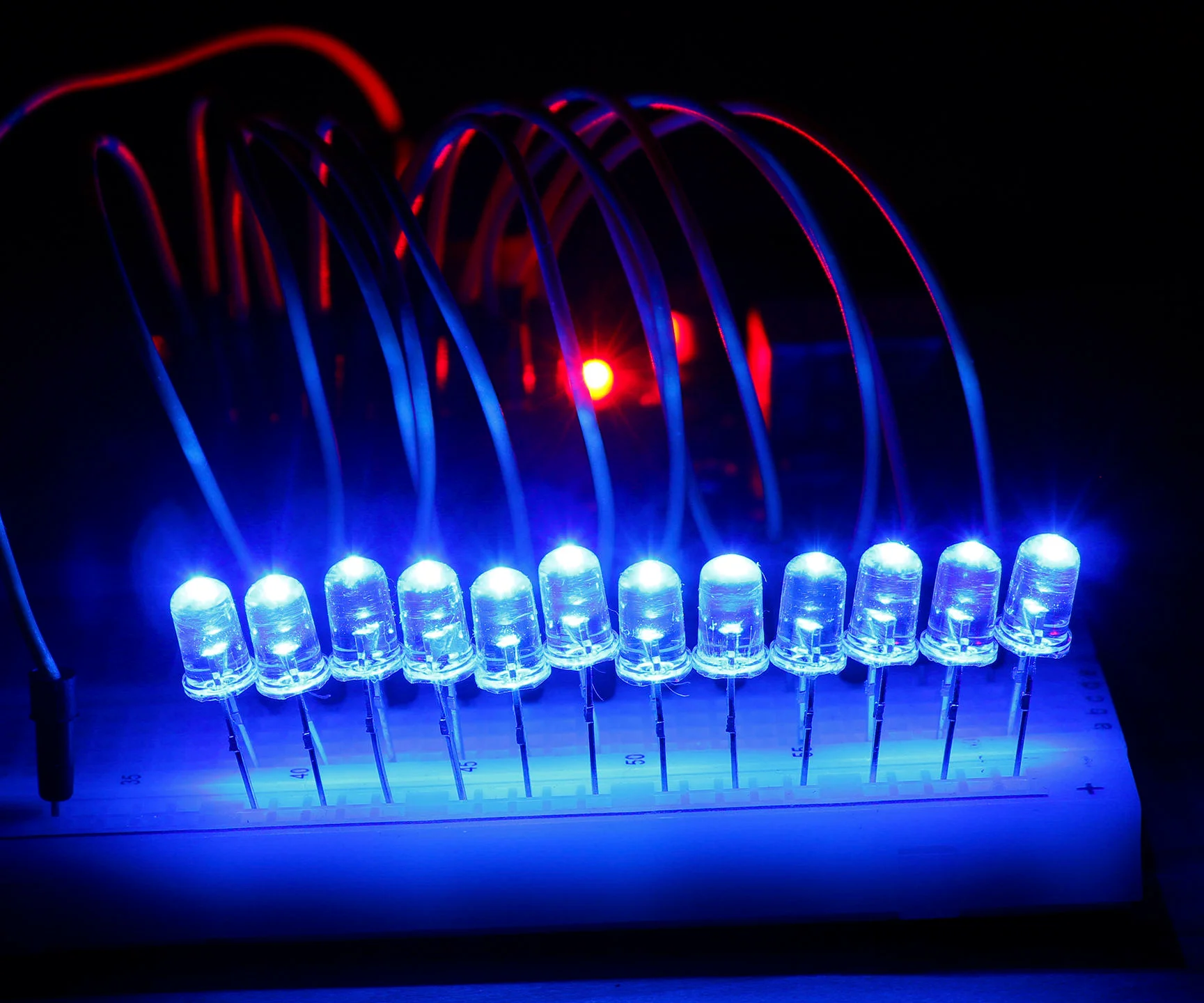
Los LEDs funcionan según el principio de la electroluminiscencia. Cuando una corriente eléctrica fluye a través de un semiconductor en el diodo, los electrones se combinan con los huecos en la estructura del semiconductor, liberando fotones de luz en el proceso. La longitud de onda de la luz emitida depende del material semiconductor y determina el color del LED.

Características Clave:

Eficiencia Energética: Los LEDs son altamente eficientes y consumen menos energía en comparación con las bombillas tradicionales, lo que los hace ideales para sistemas de monitoreo y control de huertos que pueden funcionar con energía limitada. Durabilidad: Los LEDs tienen una vida útil prolongada en comparación con las bombillas convencionales, lo que reduce la necesidad de reemplazo frecuente. Control de Intensidad: Los LEDs son fáciles de controlar en términos de intensidad lumínica, lo que permite ajustar la cantidad de luz según las necesidades específicas de las plantas en el huerto. Disponibilidad de Colores: Los LEDs están disponibles en una variedad de colores, lo que facilita la creación de condiciones de iluminación personalizadas para diferentes etapas de crecimiento de las plantas.

Aplicaciones Comunes:

* Iluminación Hortícola
* Indicadores Visuales
* Ahorro de Energía
* Control de ambiente



**Figura N°6.** LEDs en Arduino UNO

**2.6. Definición de Arduino UNO**

El Arduino Uno es una placa de microcontrolador de código abierto basado en el microchip ATmega328P y desarrollado por Arduino.cc. La placa está equipada con conjuntos de pines de E/S digitales y analógicas que pueden conectarse a varias placas de expansión y otros circuitos. La placa tiene 14 pines digitales, 6 pines analógicos y programables con el Arduino IDE (Entorno de desarrollo integrado) a través de un cable USB tipo B.

Principio de Funcionamiento:

El Arduino UNO está equipado con un microcontrolador ATmega328P que ejecuta programas escritos en el entorno de desarrollo Arduino. Los usuarios pueden programar el Arduino UNO para interactuar con sensores, actuadores y otros dispositivos electrónicos utilizando un lenguaje de programación basado en C/C++. El programa se carga en el microcontrolador a través de un cable USB y puede controlar la interacción con el entorno físico a través de los pines de entrada/salida del Arduino UNO.

Características Clave:

Facilidad de Uso: Es accesible para personas con diferentes niveles de experiencia en electrónica y programación.

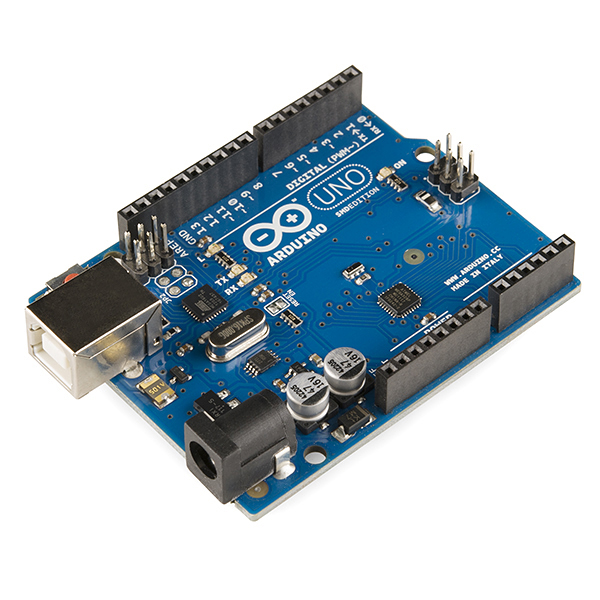
Amplia Comunidad: Existe una comunidad activa de usuarios y desarrolladores de Arduino que comparten proyectos, código y soluciones en línea, lo que facilita el aprendizaje y la resolución de problemas.

E/S Versátiles: Cuenta con pines de entrada/salida digital y analógica que permiten la conexión y compatibilidad de sensores, actuadores y otros dispositivos electrónicos.

Programación Personalizada: Puedes programar el Arduino UNO para tomar decisiones basadas en los datos recopilados por los sensores y controlar dispositivos de acuerdo con esas decisiones.

Aplicaciones Comunes:

* Monitoreo de Condiciones Ambientales
* Control de Riego
* Control de Temperatura
* Alertas y Notificaciones
* Registro de Datos



**Figura N°7.** Arduino Uno

**2.6. Definición de Protoboard**

Un protoboard, también conocido como placa de pruebas, es una placa base que se utiliza para ensamblar circuitos electrónicos sin necesidad de soldar componentes. Es una herramienta fundamental en electrónica para prototipar y probar circuitos de manera rápida y reversible.

Principio de Funcionamiento:

El protoboard consta de una serie de agujeros o puntos de conexión en los que se pueden insertar componentes electrónicos, como resistencias, sensores, cables y microcontroladores, de forma temporal y sin necesidad de soldadura. Los puntos de conexión están dispuestos en filas y columnas, y están interconectados eléctricamente en patrones específicos, lo que permite la conexión de componentes y cables para formar circuitos.

Características Clave:

Reutilización: El protoboard permite reutilizar componentes y cables en múltiples prototipos, lo que facilita la iteración y el desarrollo de nuevos circuitos sin dañar los componentes.

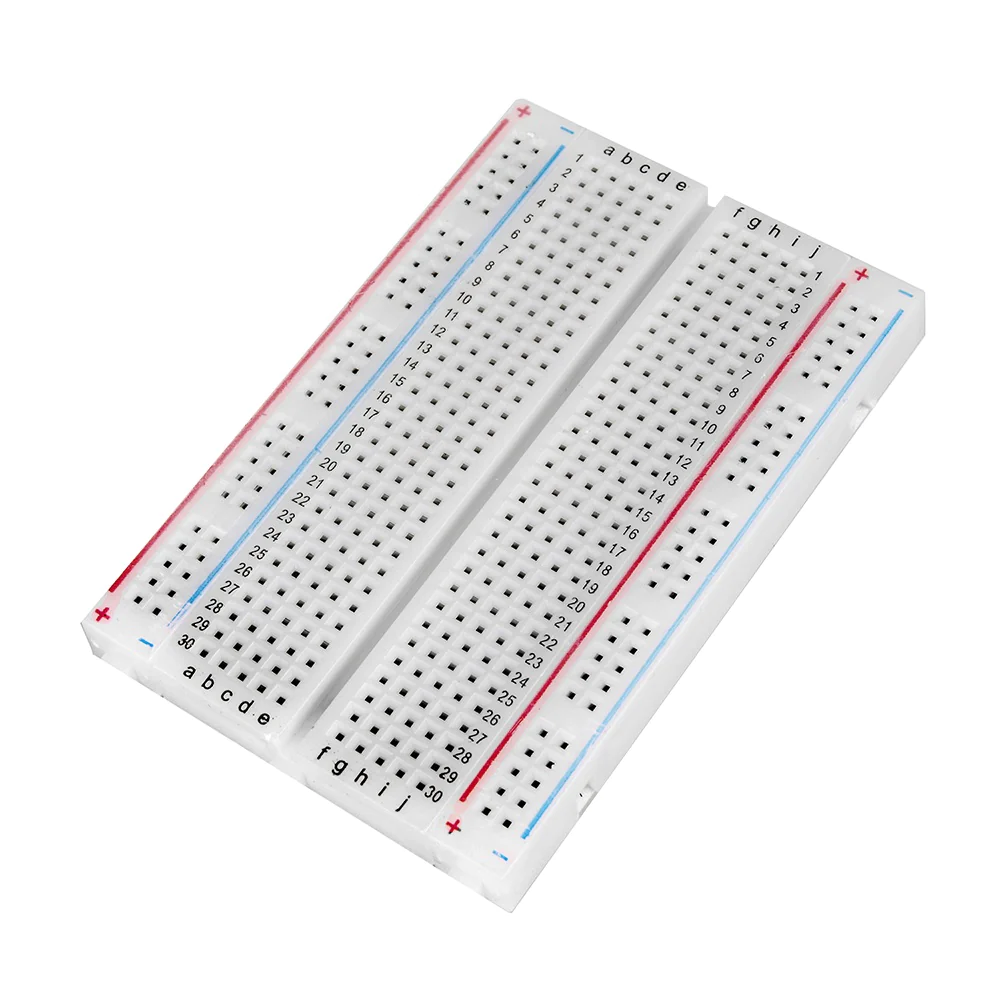
Conexiones sin soldadura: No se requiere soldadura, lo que facilita la construcción y modificación de circuitos de manera rápida y sencilla.

Organización: Los puntos de conexión están organizados en filas y columnas, lo que ayuda a mantener ordenados y estructurados los circuitos.

Facilidad de Prueba: Permite probar y depurar circuitos antes de pasar a una implementación definitiva.

Aplicaciones Comunes:

* Prototipado de Circuitos de Control
* Conexión de Sensores
* Diseño de Interfaces de Usuario
* Pruebas y Depuración
* Evaluación de Componentes



**Figura N°8.** Protoboard de 400 puntos

**2.6. Definición de Jumpers**

Los cables jumpers, también conocidos como cables puente, son cables eléctricos cortos y flexibles con conectores en ambos extremos. Se utilizan para establecer conexiones temporales entre componentes electrónicos, como sensores, placas de desarrollo (como Arduino) y otros dispositivos, sin la necesidad de soldar.

Principio de Funcionamiento:

Los cables jumpers permiten realizar conexiones eléctricas entre componentes de forma rápida y sencilla. Uno de sus extremos se conecta a un pin o conector en un componente, y el otro extremo se conecta a otro componente o ubicación deseada. Estas conexiones temporales son esenciales para la prototipación y el montaje de circuitos electrónicos en un proyecto.

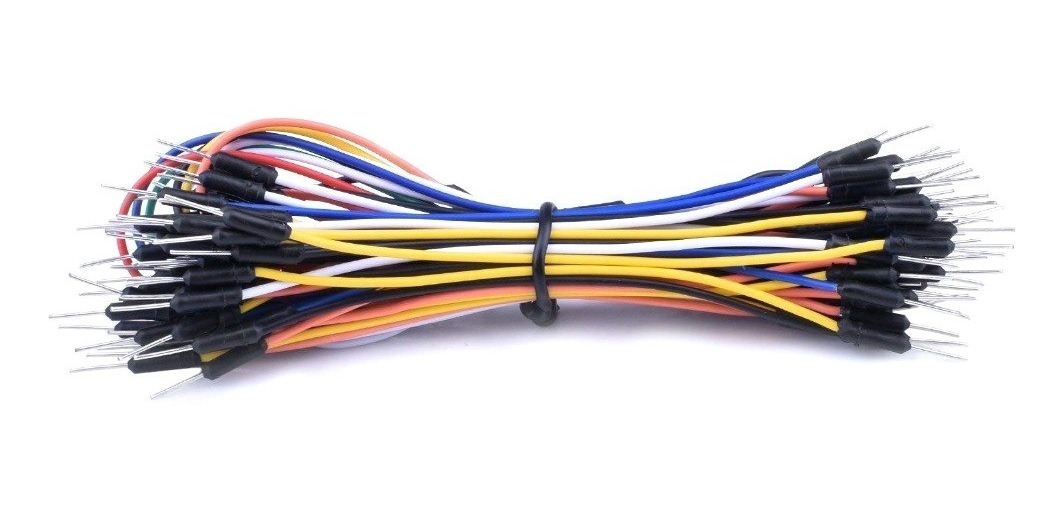
Características Clave:

Versatilidad: Los cables jumpers vienen en diferentes longitudes y colores, lo que permite una amplia flexibilidad en la conexión de componentes y en la organización de los cables en tu proyecto.

Reutilización: Son reutilizables y fáciles de desconectar y volver a conectar, lo que facilita la reconfiguración de tu sistema y la corrección de errores.

Sin soldadura: No se requiere soldadura para utilizarlos, lo que simplifica la creación y modificación de conexiones en tu proyecto.

Organización: Los cables jumpers ayudan a mantener un aspecto ordenado y organizado en tu proyecto, evitando enredos de cables y confusiones en las conexiones.



**Figura N°9.** Cables Jumpers

1. **Metodología**

**3.1. Codificación en Arduino IDE**

#include<DHT.h> //Libreria

#include<Stepper.h> //Libreria para controlar el motor

int pasosPorVuelta = 2048;

Stepper motor(pasosPorVuelta,8,10,9,11);

DHT dht(2, DHT11);

int pinSensorLluvia= A0;

int pinregar=3;

int estadolluvia = 0 ; //0 cuando no llueve y 1 cuando llueve

int estadotecho = 0 ;

int estadoregar = 0 ;

int lluvia=0;

void setup() {

// put your setup code here, to run once:

motor.setSpeed(10);

Serial.begin(9600);

pinMode(pinSensorLluvia,INPUT);

dht.begin();

}

void loop() {

int valorlluvia=analogRead(pinSensorLluvia);

float valorHumedad=dht.readHumidity();

float valorTemperatura=dht.readTemperature();

Serial.print("Humedad: ");

Serial.println(valorHumedad);

Serial.print("Valor Lluvia: ");

Serial.println(valorlluvia);

Serial.print("Temperatura: ");

Serial.println(valorTemperatura);

//verifica si está lloviendo

if(valorlluvia <900){

Serial.println("Llueve");

estadolluvia = 1;

}else{

estadolluvia=0;

}

//abrimos el techo si no esta lloviendo y está encerrado el techo

if((estadolluvia == 0)&(estadotecho == 1)){

estadotecho = 0;

motor.step(-pasosPorVuelta);

delay(2000);

}

//

if(valorHumedad < 75.00){

if(estadolluvia == 0){

estadoregar=1;

digitalWrite(pinregar,HIGH);

delay(2000);

digitalWrite(pinregar,LOW);

estadoregar=0;

}

}

if(valorHumedad>= 75.00){

if((estadolluvia == 1) & (estadotecho==0)){

Serial.print("Empezar a cerrar todo");

motor.step(pasosPorVuelta);

delay(2000);

estadotecho =1;

}

}

delay(2000);

// // put your main code here, to run repeatedly:

// lluvia=analogRead(pinSensorLluvia);

// float TemC=dht.readTemperature();

// float TemF=dht.readTemperature(true);

// float Humd=dht.readHumidity();

// //Serial.println(TemC);

// //Serial.println(lluvia);

// if(isnan(TemC) || isnan(TemF) || isnan(Humd)){

// Serial.println("Revisar la conexion");

// }

// else{

// //Serial.println("Tem:")+ String(TemC,1)+"C "+ String(TemF,1)+"F";

// //Serial.println("Hum:"+ String(Humd,1)+"%");

// Serial.print("Tem: ");

// Serial.print(TemC, 1);

// Serial.print("C ");

// Serial.print(TemF, 1);

// Serial.print("F");

// Serial.print(" | ");

// Serial.print("Hum: ");

// Serial.print(Humd, 1);

// Serial.println("%");

// }

// delay(2000);

// //if(lluvia<800){

// // Serial.println("Llueve");

// // estado=1;

// //}

// //delay(2000);

}#include<DHT.h> //Libreria

#include<Stepper.h> //Libreria para controlar el motor

int pasosPorVuelta = 2048;

Stepper motor(pasosPorVuelta,8,10,9,11); // conexion digital

DHT dht(2, DHT11);

int pinSensorLluvia= A0;

int pinregar=3; // puerto digital 3

int estadolluvia = 0 ; //0 cuando no llueve y 1 cuando llueve

int estadotecho = 0 ; // 0 cuando techo abierto y 1 cuando esta cerrado

int estadoregar = 0 ; // 0 apagado y 1 prendido

void setup() {

// put your setup code here, to run once:

motor.setSpeed(10); // velocidad del motor

Serial.begin(9600); // imprimir en consola

pinMode(pinSensorLluvia,INPUT); // recibe señal del sensor

dht.begin();

}

void loop() {

int valorlluvia=analogRead(pinSensorLluvia); // leer

float valorHumedad=dht.readHumidity(); // valor de humedad

float valorTemperatura=dht.readTemperature(); // valor de temperatura

// imprimir

Serial.print("Humedad: ");

Serial.println(valorHumedad);

Serial.print("Valor Lluvia: ");

Serial.println(valorlluvia);

Serial.print("Temperatura: ");

Serial.println(valorTemperatura);

//verifica si está lloviendo

// for infinito

if(valorlluvia <900){

Serial.println("Llueve");

estadolluvia = 1;

}else{

estadolluvia=0;

}

//abrimos el techo si no esta lloviendo y esta ecerrado el techo

if((estadolluvia == 0)&(estadotecho == 1)){

estadotecho = 0;

motor.step(-pasosPorVuelta);

delay(2000);

}

//

if(valorHumedad < 75.00){

if(estadolluvia == 0){ // si no esta lloviendo

estadoregar=1; // activar led (regar)

digitalWrite(pinregar,HIGH);

delay(2000);

digitalWrite(pinregar,LOW);

estadoregar=0;

}

}

if(valorHumedad>= 75.00){

if((estadolluvia == 1) & (estadotecho==0)){ // si esta lloviendo y el techo esta abierto

Serial.print("Empezar a cerrar todo");

motor.step(pasosPorVuelta);

delay(2000);

estadotecho =1;

}

}

delay(2000);

}

1. **Conclusiones**

El proyecto del huerto inteligente con Arduino que hemos explorado en este artículo es un ejemplo destacado de cómo la tecnología puede mejorar y simplificar nuestras vidas cotidianas.

Mediante la combinación de sensores avanzados y controladores precisos, este proyecto demuestra cómo podemos automatizar tareas, tomar decisiones más informadas y adaptarnos a las condiciones cambiantes de manera más eficiente en una amplia gama de aplicaciones. Desde la comodidad de nuestro hogar hasta proyectos más ambiciosos, el huerto inteligente de Arduino nos ofrece un vistazo al emocionante futuro de la automatización y la toma de decisiones impulsadas por datos en nuestras vidas diarias.

1. **Referencias**

*Arduino Docs | Arduino Documentation*. (s. f.). https://docs.arduino.cc/?\_gl=1\*aw9wgg\*\_ga\*MTAxNzQwODAzLjE2OTM3ODg0MTU.\*\_ga\_NEXN8H46L5\*MTY5Mzc4ODQxNC4xLjAuMTY5Mzc4ODQxNC4wLjAuMA..

Lobejon, F.G. (2022). Jardín automático monitorizado por medio de una placa Arduino. Universidad Carlos III de Madrid. Madrid. España.

https://e-archivo.uc3m.es/handle/10016/36331#preview

Bermúdez, D. (2014). Riego de huerta automatizado por Arduino. Universidad Pública de Navarra. Navarra. España. https://academica-e.unavarra.es/handle/2454/13166