Diseño de Sistema de riego y monitoreo de variables mediante IOT en los cultivos automatizado con Arduino.

Autores: Jesús Florián 23551912596
Facultad de Ingeniería Mecánica, Electrónica y Biomédica.
Tecnología en mantenimiento electromecánico industrial
Universidad Antonio Nariño
Sede Puerto Colombia
Jflorian73@uan.edu.co
Director: Rafael Antonio Ramírez Restrepo
e-mail institucional del director: rafael.ramirez@uan.edu.co

RESUMEN: se plantea diseñar un sistema de riego automático, que combine soluciones de hardware y software libres, para calcular la humedad de la tierra y el aire en vista de que estos son vitales porque forman parte en el proceso de la cosecha. A este proyecto se le añadirá un microcontrolador, que actúe como centro de operaciones para asegurar el suministro y la graduación del agua para así mantener hidratada una planta. Por lo siguiente, esta solución, incluye una aplicación móvil que utiliza tecnología Bluetooth y establece el canal de comunicación con el microcontrolador, logrando la emisión y recepción de las señales generadas por los sensores del sistema para minimizar el trabajo de las personas.

PALABRAS CLAVE*: *Automatización, recolección de datos, señales de sensores.

I. INTRODUCCIÓN

La agricultura es una actividad económica vital para el desarrollo de la humanidad, pues garantiza la producción de alimentos y materias primas para la industria. Sin embargo, una de las principales limitaciones en la producción agrícola es el suministro de agua, especialmente en zonas áridas y semiáridas. La gestión eficiente del agua es esencial para aumentar la productividad y la rentabilidad de la agricultura, y para ello se requiere de tecnologías innovadoras que permitan un uso más eficiente de este recurso.

En este contexto, se propone el diseño e implementación de un sistema de riego automatizado con Arduino para optimizar el uso del agua en la agricultura. El objetivo principal de este trabajo es reducir el consumo de agua en la producción agrícola, mediante el uso de sensores y controladores automatizados para la activación del riego en los momentos precisos y con la cantidad adecuada de agua para cada cultivo.

El sistema de riego automatizado con Arduino está pensado para su uso en huertos y cultivos pequeños, en los cuales no se cuenta con la capacidad de implementar sistemas de riego convencionales, que suelen ser costosos y complicados de instalar. El sistema consta de un conjunto de sensores para medir la humedad del suelo, la temperatura y la humedad ambiente, y un controlador que se encarga de activar la bomba de riego en función de los valores de los sensores y de la programación previa del usuario.

El método empleado en este trabajo consta de varias etapas, comenzando por la investigación y selección de los componentes electrónicos necesarios para el sistema. Luego, se procedió a la construcción del prototipo y a su prueba en condiciones reales, para evaluar su eficacia y realizar las mejoras necesarias. Finalmente, se elaboró un manual de usuario para facilitar la implementación y el uso del sistema por parte de agricultores y horticultores.

En conclusión, la implementación de un sistema de riego automatizado con Arduino es una solución tecnológica que permitirá optimizar el uso del agua en la agricultura, reducir los costos de producción y mejorar la calidad de los cultivos. Este trabajo busca contribuir al desarrollo sostenible y a la protección del medio ambiente, promoviendo una agricultura más eficiente y responsable.

II. PLANTEAMIENTO DEL PROBLEMA

Un estudio llevado a cabo por el ingeniero Federico Llatser, especialista en sistemas de riego, ha mostrado que en las zonas rurales se desperdicia un 84 por ciento del agua de riego que se capta de los ríos y arroyos debido a la falta de tecnificación en los sistemas de riego por parte de los productores. Este desperdicio de agua podría evitarse mediante un mejor manejo del recurso hídrico, lo que permitiría duplicar la producción y mejorar la calidad de los cultivos, reducir los costos de producción y recuperar tierras que antes eran improductivas debido a la falta de riego. (Clarín, p.1. 2017)

Las pérdidas de agua en los sistemas de riego se deben a la falta de conocimiento sobre las necesidades hídricas específicas de cada cultivo, así como las condiciones ambientales óptimas para su crecimiento. Cada planta y alimento presenta un comportamiento diferente, lo que requiere un manejo individualizado y adaptado a las necesidades de cada uno. La falta de este conocimiento y la ausencia de tecnología apropiada para medir y controlar el riego en función de estas variables, contribuyen a las pérdidas de agua y a la ineficiencia en la producción agrícola. Por tanto, es necesario aumentar la investigación y el desarrollo tecnológico en este ámbito para optimizar el uso del agua en la agricultura y mejorar la sostenibilidad del sector.

Claro, además, los métodos de riegos convencionales requieren tiempo y esfuerzo, lo que dificulta aumentar la producción a futuro. Muchas veces las pérdidas son por presencia de distintas plagas las cuales con el tiempo han sido identificadas y son tratadas de manera artesanal mediante múltiples técnicas. La falta de conocimiento del estado del cultivo en términos de variables ambientales no permite controlar las condiciones en la que se desarrolla la siembra, por lo que también es importante tener un acceso a la visualización de las variables en tiempo real que pueden ayudar en la toma de decisiones. Por todo lo anterior se plantea la necesidad de Diseñar un Sistema de riego y monitoreo de variables mediante el uso de la tecnología de Internet de las cosas (IOT) en los cultivos, automatizado mediante Arduino. La IOT se refiere a la interconexión de dispositivos electrónicos mediante Internet, permitiendo la transferencia de datos entre ellos y la posibilidad de controlarlos remotamente. En este caso, el sistema de riego y monitoreo de variables utilizará sensores conectados a la red IOT para recopilar datos en tiempo real sobre el estado del cultivo y el ambiente en el que se encuentra, lo que permitirá tomar decisiones informadas y precisas en cuanto a la cantidad de agua y otros recursos necesarios para optimizar la producción.

III. JUSTIFICACION

El agua es un bien esencial y valioso para las actividades agropecuarias, se requiere de un uso adecuado, teniendo cuenta la escasez en ciertas zonas del país. Actualmente con la presentación de diferentes avances tecnológicos a través de la elaboración de modelos, los sistemas de riego permiten el uso del fluido, para el consumo, sino que además garantiza la producción de calidad tanto a pequeños como a grandes niveles.

A. OBJETIVO GENERAL

- Diseñar un sistema de riego automatizado y monitoreo de variables ambientales mediante IOT en los cultivos

B. OBJETIVOS ESPECÍFICOS

- Determinar las variables ambientales para el riego adecuado de cultivos y los requerimientos de instrumentos para el sistema de control de riego y monitoreo de variables del ambiente.
- Diseñar el sistema de control, teniendo en cuenta los requisitos del terreno y del entorno al diseñar el sistema de control.
- Validar el sistema de riego automatizado para la optimización del agua y la mejora en el nivel del cultivo.
- Elaborar un procedimiento en un manual para la operación del sistema al implementarlo

IV. METODOLOGÍA

Ha sido desarrollado mediante una metodología rigurosa, que ha permitido la implementación de un prototipo funcional y eficiente. A continuación, se detallan las diferentes etapas seguidas en el desarrollo del prototipo:

1. Investigación y selección de los componentes:

   La primera etapa consistió en la investigación y selección de los componentes necesarios para el prototipo. Se realizaron estudios técnicos para definir las especificaciones de los sensores de humedad del suelo, el sensor de temperatura y humedad ambiente, el controlador Arduino, la bomba de riego, los relés y otros componentes adicionales. Se consideraron aspectos como la precisión, la sensibilidad, la resistencia y la durabilidad de los componentes.

2. Diseño del prototipo:

   Una vez seleccionados los componentes, se procedió al diseño del prototipo del sistema de riego automatizado con Arduino. Se tuvieron en cuenta las características técnicas de los cultivos a regar, la topografía del terreno y la disponibilidad de agua. Se definió la ubicación y conexión de los sensores y la bomba de riego en el terreno, y se elaboró el circuito electrónico necesario para el funcionamiento del sistema.

Ilustración 1 elaboración propia

3. Construcción y programación del prototipo:

   En esta etapa, se construyó el prototipo del sistema de riego automatizado con los componentes seleccionados. Se procedió a la programación del controlador Arduino para que active la bomba de riego en función de los valores medidos por los sensores. Se realizaron pruebas para verificar que el sistema funcionaba correctamente y se ajustó la programación y el circuito electrónico en caso de ser necesario.

4. Evaluación del prototipo en condiciones reales:

   Una vez construido y programado el prototipo, se procedió a su evaluación en condiciones reales. Se instaló el sistema en un huerto y se midió su eficacia en términos de reducción del consumo de agua, mejoras en la producción y la calidad de los cultivos. Se recopilaron datos de forma continua para evaluar el rendimiento del sistema en diferentes condiciones climáticas y de humedad del suelo.

5. Mejoras y optimización del sistema:

   En función de los resultados obtenidos en la evaluación del prototipo, se realizaron mejoras y optimizaciones al sistema. Se ajustaron los valores de los sensores, se mejoró la programación del controlador, se cambiaron algunos componentes y se reubicó el sistema para mejorar su rendimiento. Se realizaron pruebas adicionales para comprobar la eficacia de las mejoras y se recopilaron nuevos datos para evaluar el rendimiento del sistema en diferentes situaciones.

V. CAUDAL

En el proyecto del sistema de riego automatizado con Arduino, se ha implementado un procedimiento matemático para calcular el caudal de agua en el sistema, utilizando mediciones y pruebas realizadas. A continuación se detalla el procedimiento y se incluyen números de ejemplo basados en las pruebas del proyecto:

1. Medición de la presión: Se utilizó un manómetro calibrado para medir la presión del agua en el sistema. La medición arrojó un valor de presión de 30 psi (libras por pulgada cuadrada).
2. Medición del diámetro de la tubería: Se midió el diámetro interno de la tubería por la cual fluye el agua en el sistema. La medición reveló un diámetro de 2 pulgadas.

3. Determinación del coeficiente de descarga: A través de análisis y pruebas, se determinó que el coeficiente de descarga Cd para esta tubería específica es de 0.95.
4. Aplicación de la fórmula del caudal: Utilizando los datos obtenidos, se aplicó la fórmula del caudal de agua:

   Caudal = (Cd x Área de la sección transversal de la tubería x √(2gP))

   Donde:

   - Cd es el coeficiente de descarga (0.95).
   - Área de la sección transversal de la tubería se calcula utilizando la fórmula del área de un círculo: Área = π x (D/2)^2. En este caso, el diámetro D es de 2 pulgadas, por lo tanto, el área es de π x (1 pulgada)^2.
   - g es la aceleración debido a la gravedad (aproximadamente 9.81 m/s²).
   - P es la presión medida en psi convertida a unidades de fuerza por área (lb/in²).

   Aplicando los valores obtenidos en las pruebas: Área = π x (1 pulgada)^2 = 3.14 pulgadas² Caudal = (0.95 x 3.14 pulgadas² x √(2 x 9.81 m/s² x 30 psi)).

5. Conversión de unidades: El caudal se puede convertir a diferentes unidades según sea necesario. Por ejemplo, si se desea obtener el caudal en litros por minuto (L/min), se puede aplicar un factor de conversión adecuado.

VI. ESQUEMA DE VARIABLES

Ilustración 2 elaboración propia

En el esquema simplificado del proyecto, se presentan las variables que se midieron durante el proyecto, las entradas y salidas del controlador, y los valores de referencia utilizados en las acciones de control.

Variables Medidas en el Proyecto:

1. Humedad del suelo: Se midió la humedad del suelo en las zona de riego utilizando sensores de humedad del suelo.
2. Temperatura ambiente: Se midió la temperatura ambiente en el entorno del sistema de riego con sensores de temperatura.
3. Nivel de agua en el tanque: Se midió el nivel de agua en el tanque de almacenamiento utilizando sensores de nivel de agua.

Entradas del Controlador:

1. Datos de los sensores: Se utilizaron los valores medidos de humedad del suelo, temperatura ambiente y nivel de agua en el tanque como entradas al controlador.
2. Programación del calendario de riego: Se estableció un calendario de riego con horarios y duración para cada zona de riego. Estos valores se ingresaron al controlador como referencia.

Salidas del Controlador:

1. Accionamiento de la bomba de agua: El controlador activó o desactivó la bomba de agua en función de los datos de los sensores y la programación del calendario de riego.
2. Duración del riego: El controlador determinó la duración del riego en la zona de acuerdo con los valores medidos y la programación establecida.

Valores de Referencia en las Acciones de Control:

1. Umbral de humedad del suelo: Se estableció un valor de referencia de humedad del suelo que indicaba cuándo se debía iniciar el riego.
2. Punto de ajuste de temperatura: Se definió un punto de ajuste de temperatura que influyó en la frecuencia y duración del riego.
3. Nivel mínimo de agua en el tanque: Se estableció un nivel de referencia para el tanque de almacenamiento que indicaba cuándo se debía activar la recarga de agua.

Durante el proyecto realizado, se utilizaron estas variables, entradas, salidas y valores de referencia en las acciones de control del sistema de riego automatizado, logrando un riego eficiente y optimizado para las necesidades de las plantas.

VII. PRUEBAS REALIZADAS

En esta prueba nuestra planta ya desarrollada, la dejamos con nuestro prototipo cada 12 horas durante 40 días, tuvo como resultado un crecimiento de nuevas hojas.
Cabe resaltar que hubo días de lluvia en ese tiempo donde el sensor al captar la humedad por encima de lo programado en el proyecto, el riego se desactiva y reactivó al día siguiente.
Se reutilizó el agua de lluvia para regar la misma planta al siguiente día.

Ilustración 3. elaboración propia

VIII. RESULTADOS

Los resultados de las pruebas realizadas en el prototipo de sistema de riego automatizado con Arduino han sido positivos, demostrando la eficiencia del sistema en diferentes situaciones.

- El prototipo es capaz de proporcionar la cantidad de agua adecuada a los cultivos en función de sus necesidades específicas. La programación del controlador ha permitido establecer una programación de riego personalizado, en función de factores como la humedad del suelo y la temperatura. De esta manera, se ha logrado una distribución del agua evitando la falta de riego en determinadas áreas del huerto.

- Se ha comprobado que el sistema de riego permite un ahorro significativo de agua en comparación con otros sistemas de riego convencionales. La programación del controlador ha permitido ajustar la cantidad de agua suministrada a los cultivos en función de las necesidades de cada planta, evitando el despilfarro de agua y reduciendo los costos asociados.

- Se ha comprobado que el sistema es fácil de usar y mantener. La interfaz de usuario ha sido diseñada de forma intuitiva, permitiendo a los agricultores y horticultores establecer el calendario de riego y supervisar el funcionamiento del sistema de forma sencilla. Además, el mantenimiento del sistema es mínimo, ya que los componentes utilizados son de alta calidad y requieren poco mantenimiento.

- Por último, se ha comprobado que el sistema de riego automatizado con Arduino puede mejorar significativamente la producción y la calidad de los cultivos. El suministro de agua en las cantidades y momentos adecuados ha permitido mejorar el crecimiento y el desarrollo de los cultivos, lo que se ha traducido en una mayor producción y calidad de los productos cosechados.

- En conclusión, los resultados de las pruebas realizadas en el prototipo han sido muy positivos, demostrando la eficacia, la eficiencia y la viabilidad de la solución propuesta. El sistema ha demostrado ser capaz de proporcionar la cantidad de agua adecuada a los cultivos, ahorrar agua y reducir los costos asociados, ser fácil de usar y mantener, y mejorar significativamente la producción y la calidad de los cultivos.

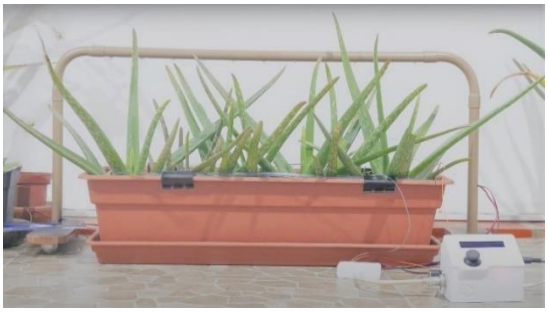


Ilustración 4. elaboración propia

IX. CONCLUSIONES

Después de haber instalado el sistema de riego automatizado por Arduino en mi jardín, he notado una mejora significativa en el cuidado y la salud de mis

plantas. El sistema me ha permitido programar de manera precisa el riego de mis plantas, lo que ha resultado en un crecimiento más saludable y eficiente. Antes de tener el sistema automatizado, regaba las plantas manualmente, pero a menudo terminan olvidando o regando demasiado o muy poco, lo que no era bueno para el crecimiento de las plantas. Con el sistema automatizado, puedo programar la cantidad y frecuencia de riego en función de las necesidades de cada planta específica, lo que ha resultado en un crecimiento más uniforme y saludable en todo el jardín.

Además, el sistema de riego automatizado por Arduino ha sido una solución eficiente para la conservación del agua. Antes, a menudo regaba en exceso las plantas, lo que resultaba en un desperdicio innecesario de agua.

Ahora, el sistema automatizado me ha permitido regar solo cuando sea necesario, lo que ha reducido significativamente el consumo de agua en el jardín.

Otra ventaja del sistema de riego automatizado por Arduino es la comodidad que ofrece. Antes, a menudo tenía que regar las plantas manualmente, lo que consumía mucho tiempo y esfuerzo. Ahora, el sistema automatizado se encarga de todo el proceso de riego, lo que me ha ahorrado tiempo y esfuerzo. También he podido ajustar y modificar la programación del sistema a medida que aprendo más sobre las necesidades de mis plantas, lo que ha sido fácil.

En general, el sistema de riego automatizado por Arduino ha sido una inversión rentable y valiosa para mejorar el aspecto y la salud de mi jardín. La precisión en la programación del riego ha mejorado el crecimiento de mis plantas y ha reducido el desperdicio de agua, mientras que la comodidad que ofrece ha ahorrado tiempo y esfuerzo.

Recomiendo encarecidamente la implementación de un sistema de riego automatizado por Arduino para cualquier persona que busque una solución práctica y eficiente para el cuidado de sus plantas y jardín.

X. BIBLIOGRAFIA

1. Agropinos. (2018). SISTEMAS DE RIEGO AGRICOLAS: CONOCE LOS TIPOS Y SU FUNCIONALIDAD. Agua, C. d. (2015). Agua. Anthura. (2016).
2. Importancia de la humedad en las plantas. Anthura. arizont. (2017).
3. El problema del agua en la agricultura. Avendaño, G. (2018). La radiografía del campo cordobés que conoció el MinAgricultura. El Tiempo. Obtenido de https://www.eltiempo.com/colombia/otrasciudades/agricultura-en-cordoba-en-cuidado-intensivos-393386 Betancur, B. (1983). Potabilización y Suministro del Agua – Decreto 2105 83
4. Dueñas, R. A. (2015). automatizacion de riego para el cultivo de flores tipo exportacion. Bogota -Colombia.
5. González Robaina, Felicita, Herrera Puebla, Julián, Hernández Barreto, Osmany, López Seijas, Teresa, & Cid Lazo, Greco. (2012). Base de datos sobre necesidades hídricas. Revista Ciencias Técnicas Agropecuarias, 21(2), 42-47. Recuperado en 27 de febrero de 2023, de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2071-00542012000200008&lng=es&tlng=pt.
6. Perez, D. O., Marceles, K., Palta, E. V., & Chanchi, G. E. G. (2019). Sistema de riego con tecnología IoT: Smart drip system. *Revista Ibérica de Sistemas e Tecnologias de Informação*, (E23), 121-133.
7. Se desperdicia el 84 por ciento del agua para riego (24 de febrero 2017), Clarín, p.1. Recuperado de:https://www.clarin.com/sociedad/desperdicia-84-ciento-agua-riego_0_rJ-xuMJZCtg.html
8. 1.- Thijs Elenbaas. (2012). Biblioteca EEprom extendida por Arduino. -, de THIJS.ELENBASS.NET Sitio web: http://thijs.elenbaas.net/2012/07/extended-eepromlibrary-for-arduino/ 2.- Novedades Agricolas SA. (2015). Precio de Riego¿Como calcularlo?. -, de NOVAGRIC Sitio web: https://www.novagric.com/es/blog/articulos/precio-riego-como-calcularlo 3.- Arduino. (2018). Arduino EEPROMex library. -, de Codebender Sitio web: https://playground.arduino.cc/Code/EEPROMex/ 4.- Luis del

XI. ANEXOS

- MATERIALES PRINCIPALES

ARDUINO NANO

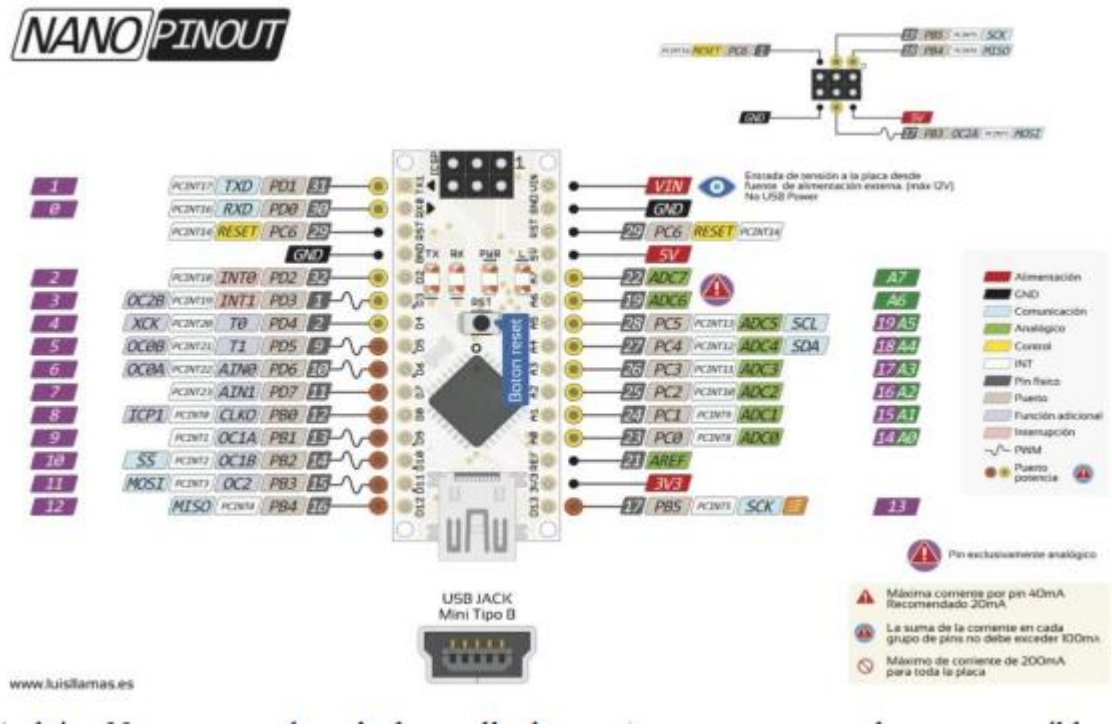


Ilustración 5

Arduino Nano es una placa de desarrollo de tamaño reducido y de bajo costo basada en el microcontrolador ATMega328P de Atmel. Es similar al popular Arduino Uno, pero es más pequeña y tiene un número reducido de pines de entrada y salida.
La placa Arduino Nano es una excelente opción para proyectos que requieren una placa compacta y liviana, como proyectos de robótica, control de motores, sistemas de sensores y aplicaciones de control de luz. La placa también es popular entre los aficionados y estudiantes que desean aprender a programar microcontroladores y electrónica.

Arduino Nano tiene características similares a otras placas Arduino, como un regulador de voltaje incorporado, conexión USB, un cristal oscilador de 16 MHz, y un conjunto de pines que permiten la conexión a sensores, actuadores y otros dispositivos electrónicos. La placa puede ser programada utilizando el entorno de desarrollo integrado (IDE) de Arduino, lo que la hace accesible incluso para principiantes en la programación y la electrónica.

PROTOBOARD

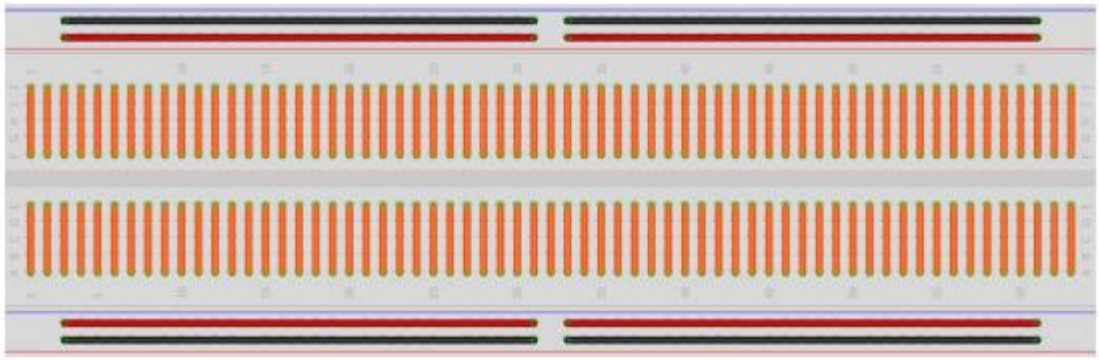


Ilustración 6

Las protoboards son muy útiles para prototipar circuitos electrónicos y para probar diferentes configuraciones de circuitos antes de soldar los componentes en una placa de circuito impreso definitiva. También son una herramienta útil para el aprendizaje de la electrónica, ya que permiten a los estudiantes experimentar con diferentes diseños de circuitos y aprender los conceptos básicos de la electricidad y la electrónica

PANTALLA LCD

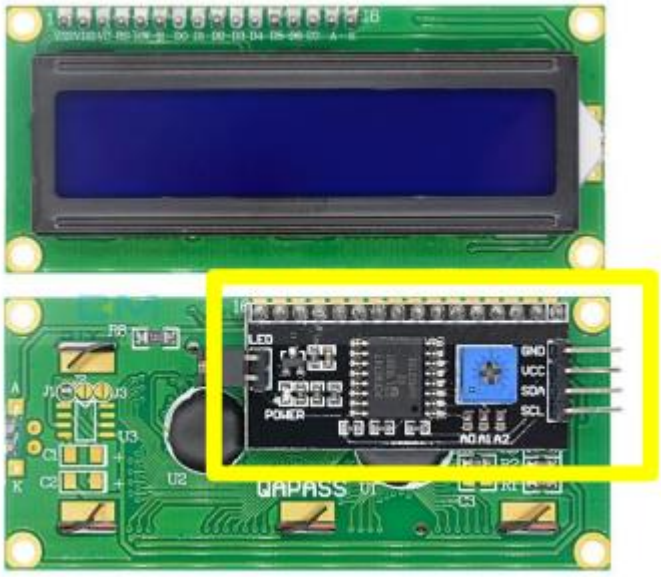


Ilustración 7

Una pantalla LCD (Liquid Crystal Display) es un tipo de pantalla que utiliza cristales líquidos para mostrar información. En el contexto de Arduino, una pantalla LCD se refiere comúnmente a una pantalla de texto que puede ser conectada a una placa Arduino para mostrar información de texto y gráficos.

JOYSTICK

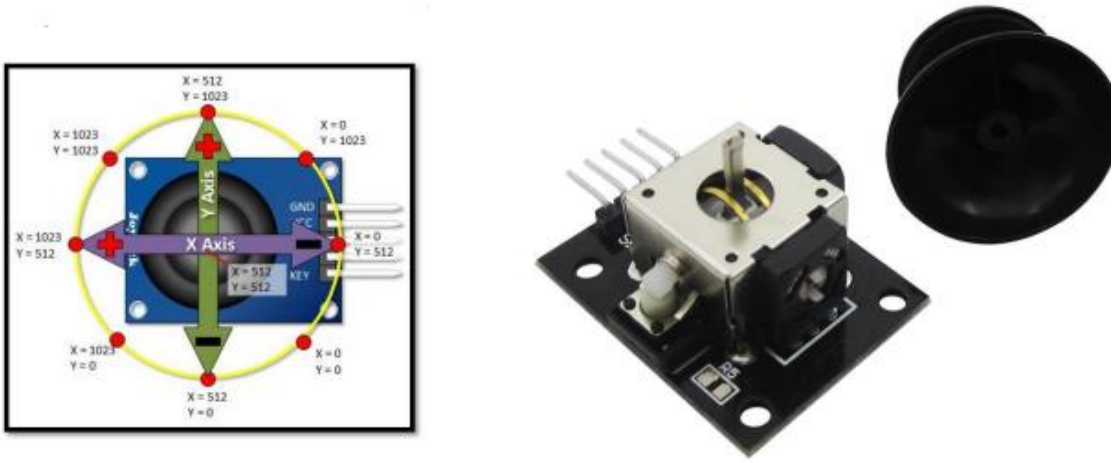


Ilustración 8

Es un dispositivo de entrada que nos permitirá navegar por los menús con mayor facilidad y comodidad.

MODULO RELAY

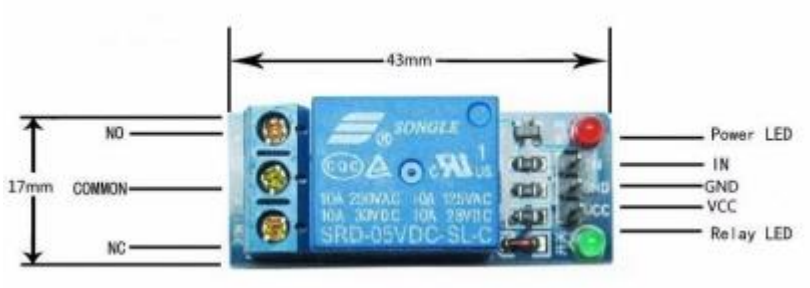


Ilustración 9

Permite controlar el encendido/apagado de equipos de alta potencia. Funciona perfectamente con Arduino.

RTC

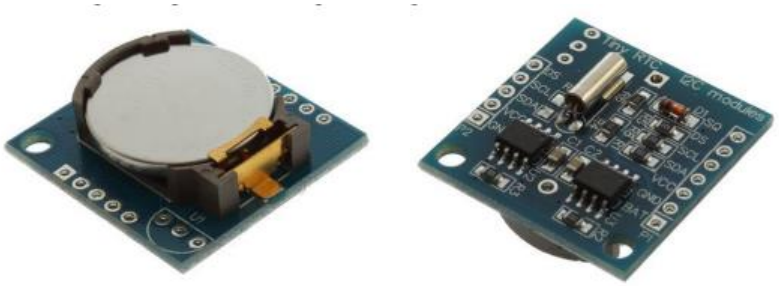


Ilustración 10

Es un reloj de un ordenador, incluido en un circuito integrado, que mantiene la hora actual. Aunque el término normalmente se refiere a dispositivos en ordenadores personales, servidores y sistemas embebidos

MODULO BLUETOOTH

Ilustración 11

Es un dispositivo que permite la comunicación inalámbrica entre una placa Arduino y otros dispositivos, como teléfonos móviles, tablets, ordenadores o cualquier otro dispositivo compatible con Bluetooth. Estos módulos son muy útiles para proyectos de robótica, domótica, automatización del hogar, monitoreo remoto, entre otros.

SENSOR DE HUMEDAD.

Ilustración 12

permite medir la cantidad de humedad presente en el aire o en un material, como el suelo o la tierra. Estos sensores son muy útiles en proyectos de jardinería, agricultura, monitoreo ambiental, entre otros.

1. PROGRAMACIÓN

- LIBRERIAS
  Wire.h
  Liquid Crystak_I2c.h:
  RTClib.h:
  EEPROMex.h/ EEPROMVar.h

2.-Scaner I2C: (Sketch)

Código para ver dirección I2C:

```
// --------------------------------------
// i2c_scanner
//
// Version 1
//    This program (or code that looks like it)
//    can be found in many places.
//    For example on the Arduino.cc forum.
//    The original author is not know.
// Version 2, Juni 2012, Using Arduino 1.0.1
//     Adapted to be as simple as possible by Arduino.cc user Krodal
// Version 3, Feb 26  2013
//    V3 by louarnold
// Version 4, March 3, 2013, Using Arduino 1.0.3
//    by Arduino.cc user Krodal.
//    Changes by louarnold removed.
//    Scanning addresses changed from 0...127 to 1...119,
//    according to the i2c scanner by Nick Gammon
//    http://www.gammon.com.au/forum/?id=10896
// Version 5, March 28, 2013
//    As version 4, but address scans now to 127.
```

```
//    A sensor seems to use address 120.
// Version 6, November 27, 2015.
//    Added waiting for the Leonardo serial communication.
//
//
// This sketch tests the standard 7-bit addresses
// Devices with higher bit address might not be seen properly.
//

#include <Wire.h>

void setup()
{
  Wire.begin();

  Serial.begin(9600);
  while (!Serial);             // Leonardo: wait for serial monitor
  Serial.println("\nI2C Scanner");
}


void loop()
{
  byte error, address;
  int nDevices;

  Serial.println("Scanning...");

  nDevices = 0;
  for(address = 1; address < 127; address++ )
  {
    // The i2c_scanner uses the return value of
    // the Write.endTransmisstion to see if
    // a device did acknowledge to the address.
    Wire.beginTransmission(address);
    error = Wire.endTransmission();

    if (error == 0)
    {
      Serial.print("I2C device found at address 0x");
      if (address<16)
        Serial.print("0");
      Serial.print(address,HEX);
      Serial.println("  !");


nDevices = 0;
for(address = 1; address < 127; address++ )
{
  // The i2c_scanner uses the return value of
  // the Write.endTransmisstion to see if
  // a device did acknowledge to the address.
  Wire.beginTransmission(address);
  error = Wire.endTransmission();

  if (error == 0)
  {
    Serial.print("I2C device found at address 0x");
    if (address<16)
      Serial.print("0");
    Serial.print(address,HEX);
    Serial.println("  !");
```

3.-Riego Automático: (Sketch)

```
/*

Programador de riego
4 riegos al día
tiempo mínimo de riego 1 minuto, máximo 99 minutos
control manual
control de la humedad de la tierra
puesta en hora de reloj
guarda los datos en la memoria EEPROM por si hay pérdida de corriente
Se pueden añadir hasta 4 sectores de riego quitando los comentarios en el código

*/

//----------------------------------------------------
// 1. Librerias
//----------------------------------------------------
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <RTClib.h>
#include <EEPROMex.h>
//#include <EEPROMVar.h>

//----------------------------------------------------
// 2. Conexiones
//----------------------------------------------------
#define xPin    A1
#define yPin    A2
#define kPin    7
#define pinS1    6
#define pinS2    5
#define pinS3    4
#define pinS4    3
#define moistureSensor A0
//SDA          A4
//SCL          A5

//----------------------------------------------------
```

```
// 3. Variables y Comandos
//-------------------------------------------------------
//leerJoystick
byte joyRead;
byte joyPos;
byte lastJoyPos;
long lastDebounceTime = 0;
long debounceDelay = 70;
int humedad;
int humedadlimite = 30;

//Control Joystick
bool PQCP;
byte editMode;

//sistema de menus
byte mNivel1;
byte nS;

//Hora
DateTime now;
int horaAc;// Hora actual en minutos (0 a (1440-1))
byte lastMinute = 0;
byte lastSecond = 0;
byte timer;

//
bool IO=1;
byte percent=100;

byte clearSave;

//
byte buffer[2];
byte nH;


byte progRec[4][9];// Tr, h1,m1...h4,m4 (for eprom)

byte controlPin[4]={pinS1,pinS2,pinS3,pinS4};
bool controlS[4];  //horario
byte TAM[4];
bool a;
//-------------------------------------------------------
// 4. Objetos
//-------------------------------------------------------
RTC_DS1307 rtc;

LiquidCrystal_I2C lcd(0x27 ,16,2);  // Atención, poner aquí la dirección i2c correcta de la pantalla LCD 16x2

//=======================================================
// SETUP
//=======================================================
void setup() {
//-------------------------------------------------------
// S1. Pines
//-------------------------------------------------------
pinMode(xPin, INPUT);
```

```
pinMode(yPin, INPUT);
pinMode(kPin, INPUT_PULLUP);
digitalWrite(pinS1,HIGH);
// digitalWrite(pinS2,HIGH);
// digitalWrite(pinS3,HIGH);
//  digitalWrite(pinS4,HIGH);
pinMode(pinS1, OUTPUT);
// pinMode(pinS2, OUTPUT);
//  pinMode(pinS3, OUTPUT);
//  pinMode(pinS4, OUTPUT);
//-------------------------------------------------------
// S2. Objetos
//-------------------------------------------------------
rtc.begin();
lcd.init();
lcd.backlight();
//-------------------------------------------------------
// S3. Program
//-------------------------------------------------------
eepromRead();
}

//=======================================================
// LOOP
//=======================================================
void loop() {x
now = rtc.now();
horaAc = now.hour()*60 + now.minute();
int sensorValue = analogRead(moistureSensor);
humedad = map(sensorValue,0,1023,99,0);

controlJoystick();

menu();

for(int i=0;i<4;i++){
//si ON y app o programa o arranque manual y humedad de la tierra es inferior al valor marcado
if ((IO&&(controlS[i]||(TAM[i]>0)))&&(humedad<humedadlimite)){
digitalWrite(controlPin[i], LOW);//off relay
} else {
digitalWrite(controlPin[i], HIGH);//on relay
}
}//for

if(now.second()!=lastSecond){
controlH();
timer++;
lastSecond=now.second();}
if (timer>30&&!(controlS[0]||controlS[1]||controlS[2]||controlS[3])&&!(TAM[0]||TAM[1]||TAM[2]||TAM[3])){
lcd.noBacklight();
}else{lcd.backlight();  }
if (timer==90){
lcd.clear();
mNivel1=0;}//vuelve a pantalla de inicio tras 90s de inactividad
if (timer>250){timer=91;}//32k

if(now.minute()!=lastMinute){
for(int i=0;i<4;i++){
```

```
if (TAM[i]>0){TAM[i]--;}
}//for
lastMinute=now.minute();}

}
void controlH(){
for(int g=0;g<4;g++){//sector
for(int i=1;i<8;i+=2){//si t!=0 y hini !=0 y horario
if((progRec[g][0]!=0)&&(((progRec[g][i]*60)+(progRec[g][i+1]))!=0)&&((horaAc>=((progRec[g][i]*60)+(progRec[g][i+1])))&&(
horaAc<((progRec[g][i]*60)+(progRec[g][i+1])+int(progRec[g][0]*(percent/100))))){
a=1;
}
}//for horas
if (a==1){controlS[g]=1;}
if (a==0){controlS[g]=0;}
a=0;
}//for sector
}

//=====================================================
// Menu
//=====================================================
void menu(){
switch (mNivel1){
case 0:
menu0();//pantalla de inicio
break;
case 1:
if(nS>0){menu11();
}else{menu1();}
break;
case 2:
menu2();
break;
case 3:
if(nS>0){menu31();
}else{menu3();}
break;
case 4:
menu4(); //
break;
case 5:
menu5(); //
break;
}//switch
}
//-----------------------------------------------------
// Pantalla de inicio
//-----------------------------------------------------
void menu0(){


lcd.setCursor(0,0);
printHour(now.hour());
printDigits(now.minute());
lcd.setCursor(7,0);
if(IO){lcd.print("PREPARADO");}
if(!IO){lcd.print("TODO  OFF");}
//  if(timer & 0x01){ par
```

```
if(timer%4==0){

lcd.setCursor(0,1);
lcd.print("HUMEDAD");
lcd.setCursor(12,1);
lcd.print(humedad);
lcd.setCursor(14,1);
lcd.print("%");



/*  lcd.setCursor(4,1);
lcd.print("S2 ");
lcd.setCursor(8,1);
lcd.print("S3 ");
lcd.setCursor(12,1);
lcd.print("S4 ") ;  */
}else{
/*   lcd.setCursor(6,1);
if (TAM[0]>0){lcd.print("mON");
}else if (controlS[0]>0){lcd.print("hON");
}else {lcd.print("OFF");}
lcd.setCursor(4,1);
if (TAM[1]>0){lcd.print("mON");
}else if (controlS[1]>0){lcd.print("hON");
}else {lcd.print("OFF");}
lcd.setCursor(8,1);
if (TAM[2]>0){lcd.print("mON");
}else if (controlS[2]>0){lcd.print("hON");
}else {lcd.print("OFF");}
lcd.setCursor(12,1);
if (TAM[0]>3){lcd.print("mON");
}else if (controlS[3]>0){lcd.print("hON");
}else {lcd.print("OFF");}*/
}
}
//-------------------------------------------------------
// MANUAL START                          Menu 1
//-------------------------------------------------------
void menu1(){
lcd.setCursor(0,0);
lcd.print("RIEGO MANUAL");
lcd.setCursor(0,1);
lcd.print("configurar -> ->");
}
//-------------------------------------------------------1.1
void menu11(){
lcd.setCursor(0,0);
lcd.print("Riego manual ");
//  lcd.print(nS);
lcd.setCursor(13,0);
if(TAM[nS-1]>0)lcd.print("ON ");
if(TAM[nS-1]==0)lcd.print("OFF");
lcd.setCursor(0,1);
lcd.print("Minutos ");
printHour(TAM[nS-1]);
}
//-------------------------------------------------------
```

```
// SET PERCENT                          Menu 2
//-------------------------------------------------------
void menu2(){
lcd.setCursor(0,0);
lcd.print("% HUMEDAD LIMITE");
lcd.setCursor(6,1);
lcd.print(humedadlimite);
if (editMode>0){
lcd.setCursor(15,1);
lcd.print("#");
} else {
lcd.setCursor(15,1);
lcd.print(" ");}
}
//-------------------------------------------------------
// SET PROGRAM                          Menu 3
//-------------------------------------------------------
void menu3(){
lcd.setCursor(0,0);
lcd.print("RIEGO AUTOMATICO");
lcd.setCursor(0,1);
lcd.print("configurar -> ->");
}
//-------------------------------------------------------3.1
void menu31(){
lcd.setCursor(0,0);
if (editMode>0){lcd.print("#");
} else {lcd.print("S");}
lcd.setCursor(1,0);
lcd.print(nS);
lcd.setCursor(0,1);
lcd.print("T ");//tr
printHour(progRec[nS-1][0]);//tr
lcd.setCursor(5,0);
printHour(progRec[nS-1][1]);//h1
lcd.setCursor(7,0);
printDigits(progRec[nS-1][2]);
lcd.setCursor(11,0);
printHour(progRec[nS-1][3]);//h2
lcd.setCursor(13,0);
printDigits(progRec[nS-1][4]);
lcd.setCursor(5,1);
printHour(progRec[nS-1][5]);//h3
lcd.setCursor(7,1);
printDigits(progRec[nS-1][6]);
lcd.setCursor(11,1);
printHour(progRec[nS-1][7]);//h4
lcd.setCursor(13,1);
printDigits(progRec[nS-1][8]);
}
//-------------------------------------------------------
// Clear / Save EEPROM                  Menu 4
//-------------------------------------------------------
void menu4(){
lcd.setCursor(4,0);
lcd.print("EEPROM");
if (editMode){
lcd.setCursor(5,1);
```

```
if (clearSave==0){
lcd.print("Cancelar");}
if (clearSave==1){
lcd.print("Guardar");}
if (clearSave==2){
lcd.print("Borrar");}
lcd.setCursor(15,1);
lcd.print("#");
} else {
lcd.setCursor(1,1);
lcd.print("Borrar/Guardar");
lcd.setCursor(15,1);
lcd.print(" ");}
}
//--------------------------------------------------
void progRecClear(){
for(int a=0;a<4;a++){
for(int b=0;b<8;b++){
progRec[a][b]=0;
}
}
}
//--------------------------------------------------
void eepromWrite(){
int address=0;
for(int i=0;i<4;i++){
for(int j=0;j<8;j++){
EEPROM.writeByte(address, progRec[i][j]);
address++;
}
}
}
//--------------------------------------------------
void eepromRead(){
int address=0;
for(int i=0;i<4;i++){
for(int j=0;j<8;j++){
progRec[i][j]=EEPROM.readByte(address);
address++;
}
}
}
//--------------------------------------------------
// Set HORA                          Menu 5
//--------------------------------------------------
void menu5(){
lcd.setCursor(0,0);
lcd.print("Cambia hora");
if (editMode>0){
lcd.setCursor(10,0);
lcd.print("#");}
lcd.setCursor(11,0);
printHour(buffer[0]);
printDigits(buffer[1 ]);
lcd.setCursor(0,1);
lcd.print("Actual  ");
lcd.setCursor(11,1);
printHour(now.hour());
```

```
printDigits(now.minute());
}
//=====================================================
// Control Joystic
//=====================================================
void controlJoystick(){
leeJoystick();
if(PQCP) {
PQCP=0;
timer=0;
//-----------------------------------------------------
// JOYSTICK BUTTON== joyPos=5
//-----------------------------------------------------
if (joyPos==5&&mNivel1==0){IO=!IO;
//manual start
//percent
}else if (joyPos==5&&mNivel1==2&&editMode==0){
editMode=2;
}else if (joyPos==5&&mNivel1==2&&editMode>0){
editMode=0;

//set prog
}else if (joyPos==5&&mNivel1==3&&nS>0&&editMode==0){
editMode=3;
nH=0;

}else if (joyPos==5&&mNivel1==3&&editMode>0){//on exit
editMode=0;

//eprom
}else if (joyPos==5&&mNivel1==4&&editMode==0){
editMode=4;
lcd.clear();
}else if (joyPos==5&&mNivel1==4&&editMode>0){//on exit
if (clearSave==1){//save
eepromWrite();
}
if (clearSave==2){//clear
progRecClear();
eepromWrite();
//syncProg();
}
editMode=0;

//sethora
}else if (joyPos==5&&mNivel1==5&&editMode==0){
editMode=5;
nH=0;
buffer[0]=now.hour();
buffer[1]=now.minute();
}else if (joyPos==5&&mNivel1==5&&editMode>0){//on exit
nH=0;
DateTime dt(2015, 1, 1, buffer[0], buffer[1], 0);
rtc.adjust(dt);
editMode=0;
}
//-----------------------------------------------------
//OTROS CONTROLES
```

```
//-------------------------------------------------------
switch (editMode){
case 0:         //no editMode
if (mNivel1==1&&joyPos==3&&TAM[nS-1]>0&&nS>0){
TAM[nS-1]--;}
if (mNivel1<5&&joyPos==3&&(mNivel1!=1||nS==0)){  //UP
lcd.clear();
mNivel1++;
nS=0;}
if (mNivel1==1&&joyPos==4&&TAM[nS-1]<99&&nS>0){    //DOWN
TAM[nS-1]++;}
if (mNivel1>0&&joyPos==4&&(mNivel1!=1||nS==0)){
lcd.clear();
mNivel1--;
nS=0;}
if ((mNivel1==1||mNivel1==3)&&joyPos==1&&nS<1){  //RIGHT
lcd.clear();
nS++;}
if ((mNivel1==1||mNivel1==3)&&joyPos==2&&nS>0){ //LEFT
lcd.clear();
nS--;}
case 1: // editMode Manual start
break;
case 2: // editMode Ser Percent
if (joyPos==4&&humedadlimite<99){//UP
lcd.clear();
humedadlimite+=10;}
if (joyPos==3&&humedadlimite>1){//DOWN
lcd.clear();
humedadlimite-=10;}
break;
case 3: // editMode Set Program
//if (joyPos==3&&buffer[nH]>0){buffer[nH]--;}  //abajo*/
if (joyPos==4&&nH==0&&progRec[nS-1][0]<99){progRec[nS-1][0]++; } //arriba
if (joyPos==4&&(nH==1||nH==3||nH==5||nH==7)&&progRec[nS-1][nH]<23){
progRec[nS-1][nH]++; }//arriba
if (joyPos==4&&(nH==2||nH==4||nH==6||nH==8)&&progRec[nS-1][nH]<59){
progRec[nS-1][nH]++; } //arriba
if (joyPos==3&&progRec[nS-1][nH]>0){progRec[nS-1][nH]--;}  //abajo
if (joyPos==1&&nH<8){nH++;  }     //derecha
if (joyPos==2&&nH>0){nH--;  }     //izq
break;
case 4: // editMode Save/clear Proram
if (joyPos==3&&clearSave<2){//DOWN
lcd.clear();
clearSave++;}
if (joyPos==4&&clearSave>0){//UP
lcd.clear();
clearSave--;}
break;
case 5: // editMode Set Time
if (joyPos==4&&buffer[0]<23&&nH==0){
buffer[0]++; }//arriba
if (joyPos==4&&buffer[1]<59&&nH==1){
buffer[1]++; } //arriba
if (joyPos==3&&buffer[nH]>0){buffer[nH]--;}  //abajo
if (joyPos==1&&nH<1){nH++;  }     //derecha
if (joyPos==2&&nH>0){nH--;  }     //izq
```

```
break;
}//!edit
}//PQCP
}
//------------------------------------------------------
int leeJoystick(){
//------------------------------------------------------
int x = analogRead(xPin);
int y = analogRead(yPin);
int k = digitalRead(kPin);
if(x>900){joyRead=1;      //x+
}else if(x<100){joyRead=2;  //x-
}else if(y>900){joyRead=3;  //y+
}else if(y<100){joyRead=4;  //y-
}else if(!k){joyRead=5;
}else{joyRead=0;}

if (joyRead != lastJoyPos){lastDebounceTime = millis();}
if(((millis() - lastDebounceTime) > debounceDelay)&&(joyRead!=joyPos)){
joyPos=joyRead;
if(!PQCP){PQCP=1;}
}
if(((millis() - lastDebounceTime) > (4*debounceDelay))&&(joyPos==3||joyPos==4)){
joyPos=joyRead;                //repeat time
if(!PQCP){PQCP=1;}
}
lastJoyPos=joyRead;
}
////////////////////////////////////imprime horas
void printHour(byte digits){
if(digits < 10){
lcd.print(' ');
lcd.print(digits,DEC);}
else {lcd.print(digits,DEC);}
}
////////////////////////////////////imprime minutos :00
void printDigits(byte digits){
lcd.print(":");
if(digits < 10){
lcd.print('0');
lcd.print(digits,DEC);}
else {lcd.print(digits,DEC);}
}
```

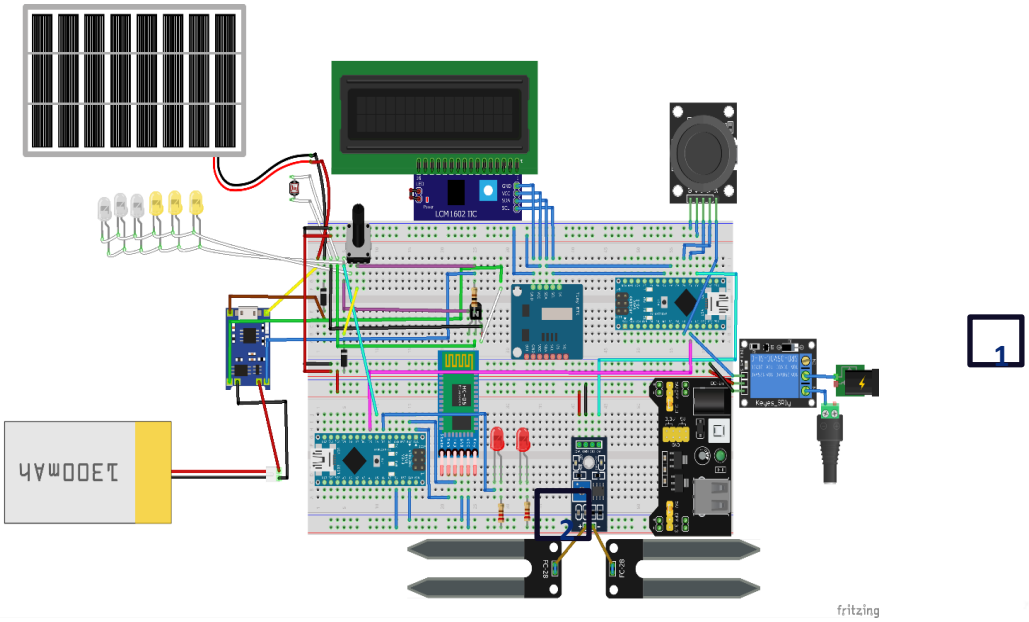
4.-Código Humedad Bluetooth (Sketch)

```
/*
 Arduino's Bluetooth with Android


 */
const int ledRed = 5;      // Terminal donde conectamos nuestro sistema de Luz
const int ledGreen = 3;    // Terminal donde conectamos nuestra Bomba de Agua
const int ledBlue = 4;     // Terminal Libre/ Variable

const char separadorDatos = '#'; // Carácter que indica que es un Dato lo que enviamos

boolean statusLedRed = true;
boolean statusLedGreen = true;
boolean statusLedBlue = true;

void setup() {
```

```
  // Configuramos las terminales donde están conectados los leds como salidas
  pinMode(ledRed, OUTPUT);
  pinMode(ledGreen, OUTPUT);
  pinMode(ledBlue, OUTPUT);

  // Inicializamos los leds y deben de prenderse
  digitalWrite(ledRed, statusLedRed);
  digitalWrite(ledGreen, statusLedGreen);
  digitalWrite(ledBlue, statusLedBlue);

  // Inicializamos el puerto serial con una velocidad de 9600
  Serial.begin(9600);
}

// Este metodo se ejecuta infinitas veces
void loop() {
  // Agregamos el caracter # para indicar que es un dato
  Serial.print("#");

  // Leemos la entrada analógica A0 y enviamos el resultado por el puerto serial
  Serial.println(analogRead(A0));

  // Esperamos un tiempo de 20 milisegundos
  delay(200);
}

/*La función SerialEvent se ejecuta cuando un nuevo dato
  llega al hardware por la terminal Rx*/
void serialEvent() {
  // Si esta disponible el puerto serial leemos los datos
  while (Serial.available()) {

    // Obtiene el siguiente byte que se recibió, este es un carácter
    char comando = (char)Serial.read();

    // Dependiendo del carácter recibido ejecutamos una acción
    switch (comando) {
      case 'r':
        cambiarLedRojo();
        break;
      case 'a':
        cambiarLedAzul();
        break;
      case 'v':
        cambiarLedVerde();
        break;
      default:
        // Si no es ningún carácter de comando regresamos el siguiente mensaje
        Serial.print(comando);
        Serial.println(": Comando no reconocido");
        break;
    }
  }
}

void cambiarLedRojo() {
  // Cambiamos el estado del LED
  statusLedRed = !statusLedRed;
  digitalWrite(ledRed, statusLedRed);
  Serial.println("Led Rojo cambio");
}

void cambiarLedAzul() {
  // Cambiamos el estado del LED
  statusLedBlue = !statusLedBlue;
  digitalWrite(ledBlue, statusLedBlue);
  Serial.println("Led Azul cambio");
}
```

```
void cambiarLedVerde() {
  // Cambiamos el estado del LED
  statusLedGreen = !statusLedGreen;
  digitalWrite(ledGreen, statusLedGreen);
  Serial.println("Led Verde cambio");
}
```

A. PROCEDIMIENTO



Bomba de Agua

Ilustración 13

2.-Las pruebas: En este paso probamos que todos los componentes emisores de luz se enciendan al conectarle alimentación y con la luz apagada para ver los led con mayor claridad. En caso que no pase eso debemos verificar que no haya un corto y que las conexiones de alimentación no tengan contacto. 3.-Ttransfernecia de la programación: Conectamos el Arduino Nano(1) a una computadora, Luego subimos el código Scaner I2c para ver las direcciones del módulo serial de la plataforma Arduino. Una vez anotado la dirección del módulo, abrimos es sketch de Programación de Riego y cambiamos la dirección de este por la que nos dio el código anterior. Después subimos es sketch al Arduino Nano(1) 4.-Subimos el sketch Humedad Bluetooth al Arduino Nano(2). 5.-Conectar el módulo Bluetooth HC-06.

B. FUNCIONAMIENTO

Su funcionamiento es a base de diferentes menús.

MENU 1:

Se configura a que hora y por cuantos minutos será regada nuestra planta

Ilustración 14. Elaborado por el autor

Ilustración 15. Elaborado por el autor

MENU 2:

Establecemos la humedad necesaria de nuestra planta.

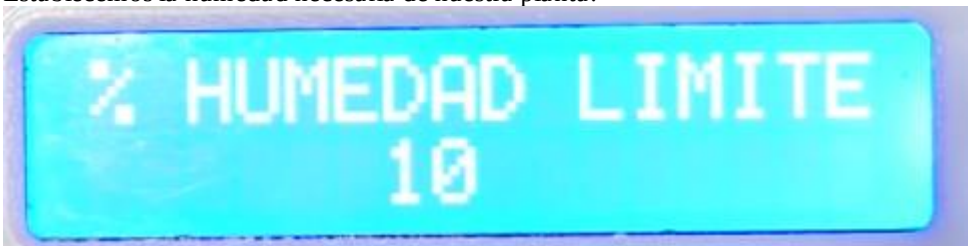


Ilustración 16. Elaborado por el autor

MENU 3:
Tenemos la posibilidad de regar nuestra plata manualmente por minutos si son sus requerimientos

Ilustración 17. Elaborado por el autor