



Projeto de Controle de Umidade e Irrigação Inteligente

Aginaldo Donizete Pires, Prof. Leandro Carlos Fernandes

¹ Faculdade de Computação e Informática
Universidade Presbiteriana Mackenzie (UPM) – São Paulo, SP – Brazil

10370027@mackenzista.com.br

Abstract. *This article presents the development and implementation of a soil moisture and irrigation control system aimed at optimizing water management in agricultural environments. The system employs soil moisture sensors to continuously monitor soil moisture conditions. Based on these readings, an automated controller activates irrigation systems as needed to maintain optimal soil moisture levels. Practical deployment of the system has shown a significant improvement in water efficiency, leading to water savings. Furthermore, it is highlighted that the system will be demonstrated in a simplified manner in a pilot plant but can be scaled according to the requirements of larger agricultural areas.*

Resumo. *Este artigo apresenta o desenvolvimento e implementação de um sistema de controle de umidade e irrigação, projetado para otimizar a gestão hídrica em ambientes agrícolas. O sistema utiliza sensor de umidade do solo para monitorar continuamente as condições de umidade no solo. Com base nessa leitura, um controlador automatizado aciona os sistemas de irrigação conforme necessário para manter níveis ideais de umidade no solo. A Implantação prática do sistema demonstrou uma melhoria significativa na eficiência hídrica, resultando em economia de água. Além disso, destaca-se que o sistema será apresentado de forma simples em uma planta-piloto, mas pode ser escalado conforme a necessidade de áreas agrícolas maiores.*

1. Introdução

A gestão eficaz da água é crucial para a sustentabilidade e produtividade dos sistemas agrícolas. Nesse contexto, o desenvolvimento de sistemas de controle de umidade e irrigação tem sido objeto de crescente interesse e pesquisa. Este artigo apresenta o desenvolvimento e implementação de um sistema de controle de umidade e irrigação projetado para otimizar a gestão hídrica em ambientes agrícolas.


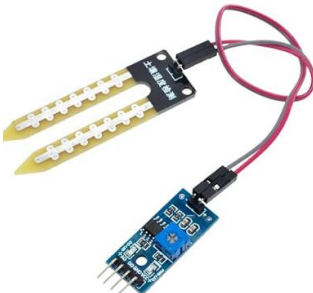
Historicamente, a gestão da umidade do solo e irrigação agrícola confiava em métodos empíricos e sistemas tradicionais. Com o avanço da tecnologia, surgiram sistemas mais avançados, como irrigação por gotejamento e aspersão, permitindo uma aplicação mais precisa da água.



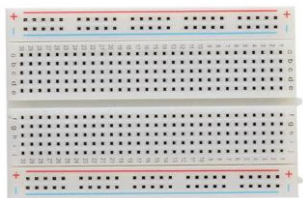



No campo da automação agrícola, estudos anteriores exploraram o uso de sensores de umidade do solo e controladores automatizados para otimizar a irrigação. Embora tenham demonstrado benefícios na economia de água e aumento da produtividade, muitos desses sistemas eram complexos e exigiam expertise técnica.

Trabalhos relacionados abordaram o desenvolvimento de sistemas semelhantes, como o estudo de Smith et al. (2018), que propôs um sistema de irrigação baseado em sensores de umidade do solo e algoritmos de controle inteligente. No entanto, há uma lacuna na disponibilidade de sistemas acessíveis, simples de implementar e escaláveis para atender às necessidades de agricultores de diferentes contextos e escalas de produção. Este trabalho busca preencher essa lacuna, apresentando um sistema de controle de umidade e irrigação que combina eficiência, praticidade e adaptabilidade para aplicação em diversos ambientes agrícolas.

2. Materiais e métodos

2.1 Lista de Materiais

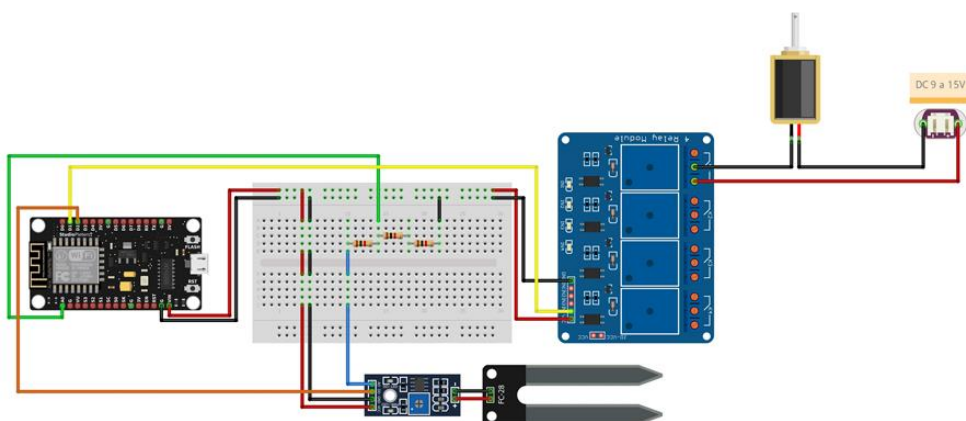
Item	Descrição	Imagem	Fonte
1	ESP8266: é um chip de microcontrolador Wi-Fi altamente integrado e de baixo custo, que se tornou extremamente popular na comunidade de desenvolvimento de projetos de IoT (Internet das Coisas). Ele é fabricado pela Espressif Systems e oferece uma plataforma poderosa para criar uma variedade de dispositivos conectados à internet.		https://www.auscomtech.com.au/product/nodemcu-lua-wireless-iot-wifi-internet-development-board-based-esp8266-cp2102/
2	Sensor de Umidade		https://www.amazon.com.br/M%C3%B3dulo-Sensor-Umidade-Solo-Higr%C3%B4metro/dp/B0C9VRY9F6

3	Fonte Alimentação 9v		https://www.eletrogate.com/fonte-9v-1a-bivolt-para-arduino
4	Minibomba de água		https://www.circuitofacil.com.br/produto/mini-bomba-6v-a-12v-agua-ar-vacuo-aquario-rs-385/
5	Protoboard		https://www.marinostore.com/acessorios/protoboard-400-pontos
6	Modulo Relé 2 canais 5v		https://www.circuitofacil.com.br/produto/modulo-rele-2-canais-5v/
7	Resistores 1k (3unidades)		https://www.usinainfo.com.br/resistor-14-watt/resistor-1k-14w-kit-com-10-unidades-2974.html
8	Placa de expansão ESP8266 30 pinos		https://www.autocorerobotica.com.br/modulo-de-expansao-para-devkit-esp32-30-pinos

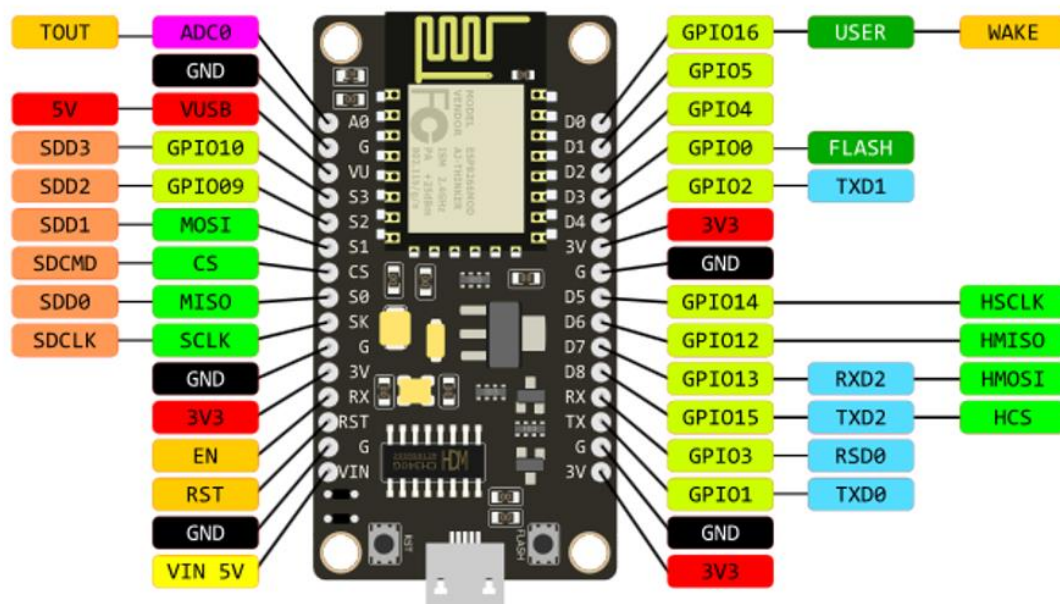
2.2 Métodos

1. **Montagem do Circuito:** Montagem dos componentes eletrônicos em uma protoboard ou em uma placa de circuito impresso (PCB). Conectando o ESP8266, o sensor de umidade do solo, a válvula solenoide, o relé e outros componentes conforme necessário.
2. **Conexão com a Rede Wi-Fi:** Conectar ESP8266 à uma rede Wi-Fi local. Isso permitirá que o ESP8266 se comunique com outros dispositivos e serviços na sua rede.
3. **Calibração do Sensor de Umidade do Solo:** Calibragem do sensor de umidade do solo para garantir leituras precisas da umidade do solo.
4. **Desenvolvimento do Software de Controle:** Código necessário para o ESP8266 ler os dados do sensor de umidade do solo e controlar a válvula solenoide ou a bomba de água com base nessas leituras. Você pode usar a IDE do Arduino ou uma plataforma de desenvolvimento similar para programar o ESP8266.
5. **Implementação de Lógica de Controle:** elaborar e implementar a lógica de controle para decidir quando ligar e desligar a irrigação com base nos dados do sensor de umidade do solo. Você pode definir limites de umidade para ativar a irrigação e evitar excesso de umidade.
6. **Testes e Ajustes:** Teste do sistema em diferentes condições e ajuste a lógica de controle conforme necessário para garantir um funcionamento confiável.
7. **Integração com Plataforma IoT:** Monitoramento e controle do sistema remotamente foi integrando com a plataforma IoT (Internet das Coisas) Mosquitto que é um broker MQTT (Message Queuing Telemetry Transport) de código aberto que permite a comunicação entre dispositivos IoT de forma eficiente e em tempo real. MQTT é um protocolo de comunicação leve, ideal para dispositivos que possuem recursos limitados, como sensores e atuadores, comuns em aplicações IoT.

2.3 Circuito



2.4 Pinagem ESP8266



2.5 Funcionamento

- 1) Primeiro será conectado dois jumpers no sensor e conectar a protoboard, no negativo e positivo;
- 2) O pino GND da protoboard deve ser conectado ao pino G do ESP8622;
- 3) O pino VCC da protoboard deve ser conectado ao pino VIN do ESP8622;
- 4) O pino D0 do sensor de umidade deve ser conectado ao Pío D2 do ESP8622;
- 5) O Pino A0 do sensor fornece uma tensão de 5V, como o ESP8622 suporta somente 3.3V é necessário fazer um regulador de tensão, para isso vamos utilizar 3 resistores de 1K Ω em série, dessa forma a tensão de saída será de 3.3V;
- 6) O Pino A0 do sensor deve ser conectado ao primeiro resistor (R1);
- 7) O segundo resistor (R2) precisa estar ligado ao pino A0 do ESP8622;
- 8) O terceiro resistor (R3) deve ser ligado ao GND do circuito;
- 9) É necessário conectar Vin do circuito no pino VCC do Relé;
- 10) O GND do circuito será conectado no pino GND do Módulo Relé;
- 11) O Pino Digital D1 do ESP8622 deve estar no IN1 do Relé;
- 12) Um terminal da Minibomba de estar conectado no contato aberto do relé K1 e outro em uma fonte de 9V;
- 13) O contato fechado do relé K1 deve estar conectado diretamente na fonte de 9 a 15V;
- 14) Nesse projeto, também vamos utilizar o aplicativo Blynk, para nos avisar quando a umidade da terra está baixa e a bomba será ligada.

3. Implementação Protocolo MQTT

Nesse projeto optei pela implementação do protocolo MQTT usando a seguinte tecnologia:

- IDE: Arduino
- Linguagem de Programação: Baseada em C/C++
- Bibliotecas: ESP8266WiFi e PubSubClient
- Ferramenta Conexão: MQTT Box
- Broker Publico: test.mosquitto.org

A implementação foi realizada em 3 etapas:

Etapla 1: Configuração do Broker MQTT na Nuvem

Configuração do broker público Mosquitto em 'test.mosquitto.org'. Este é um broker gratuito e ideal para testes.

Etapla 2: Programação do ESP8266

1. Configurar a Arduino IDE para o ESP8266:
 - Adicione a URL do gerenciador de placas do ESP8266 nas preferências da Arduino IDE:
`http://arduino.esp8266.com/stable/package_esp8266com_index.json`
 - Vá para **Ferramentas > Placa > Gerenciador de Placas**, procure por "esp8266" e instale.
2. **Instalar a Biblioteca PubSubClient:**
 - a. Vá para **Sketch > Incluir Biblioteca > Gerenciar Bibliotecas**, procure por "PubSubClient" e instale.

3. Código para o ESP8266:

```
#include <ESP8266WiFi.h>
```

```
#include <PubSubClient.h>
```

```
// Definindo os pinos
```

```
int PinoAnalogico = A0; // Define o pino A0 como Pino Analógico do sensor
int PinoDigital = D2; // Define pino D2 como Pino Digital do Sensor
int Rele = D1; // Pino Digital D1 como Relé
```

```
int EstadoSensor = 0;
int UltimoEstSensor = 0;
int ValAnalogIn; // Introduz o valor analógico ao código
```

```
// Configurações da rede WiFi
const char* ssid = "XXXXXX"; // Substitua pelo nome da sua rede WiFi
const char* password = "XXXXXX"; // Substitua pela senha da sua rede WiFi
```

```
// Configurações do MQTT
const char* mqtt_server = "test.mosquitto.org";
WiFiClient espClient;
PubSubClient client(espClient);
```

```
void setup_wifi() {
    delay(10);
    Serial.println();
    Serial.print("Conectando-se a ");
    Serial.println(ssid);

    WiFi.begin(ssid, password);

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
}
```

```
Serial.println("");
Serial.println("WiFi conectado");
Serial.println("Endereço IP: ");
Serial.println(WiFi.localIP());
```

```
}
```

```
void reconnect() {  
  while (!client.connected()) {  
    Serial.print("Tentando conexão MQTT...");  
    if (client.connect("test.mosquitto.org")) {  
      Serial.println("conectado");  
      // Não precisamos subscrever a nenhum tópico neste caso  
    } else {  
      Serial.print("falhou, rc=");  
      Serial.print(client.state());  
      Serial.println(" tentando novamente em 5 segundos");  
      delay(5000);  
    }  
  }  
}
```

```
void setup() {  
  Serial.begin(9600);  
  
  pinMode(Rele, OUTPUT); // Declara o Rele como Saída Digital  
  pinMode(PinoDigital, INPUT); // Declara o pino do sensor como Entrada Digital  
  
  setup_wifi();  
  client.setServer(mqtt_server, 1883);  
}
```

```
void loop() {  
  if (!client.connected()) {  
    reconnect();  
  }  
  client.loop();  
  
  // Lê o estado do sensor digital
```



```

EstadoSensor = digitalRead(PinoDigital);

if (EstadoSensor == 1 && UltimoEstSensor == 0) {
    Serial.println("Umidade Baixa: Irrigando a Planta");
    client.publish("estado_irrigacao", "Umidade Baixa: Irrigando a Planta");
    UltimoEstSensor = 1;
    delay(1000);
}
else if (EstadoSensor == 1 && UltimoEstSensor == 1) {
    delay(1000);
}
else {
    UltimoEstSensor = 0;
    delay(1000);
}

// Lê o valor do sensor analógico
ValAnalogIn = analogRead(PinoAnalogico);
int Porcento = map(ValAnalogIn, 1023, 0, 0, 100); // Transforma o valor analógico em
percentagem

// Exibe os valores no monitor serial
Serial.print("Umidade: ");
Serial.print(Porcento);
Serial.println("%");

// Publica o valor da umidade no tópico MQTT
String payload = String(Porcento);
client.publish("umidade", payload.c_str());

// Verifica o nível de umidade e aciona o relé se necessário
if (Porcento <= 25) { // Se a porcentagem for menor ou igual a 25%
    Serial.println("Irrigando Planta");
    client.publish("estado_irrigacao", "Irrigando Planta");
}

```

```
digitalWrite(Relé, HIGH); // Aciona Relé (liga a bomba)
}
else {
  Serial.println("Planta Irrigada");
  client.publish("estado_irrigacao", "Planta Irrigada");
  digitalWrite(Relé, LOW); // Desliga Relé (desliga a bomba)
}

delay(5000); // Adiciona um atraso de 5 segundos antes de ler os valores novamente
}
```

Etapa 3: Configuração do MQTT Box

1. **Instalação do MQTT Box:** Baixe e instale o MQTT Box do site oficial ou da loja de aplicativos do seu sistema operacional.
2. **Configuração de uma Conexão MQTT:**
 - Abra o MQTT Box e crie uma nova conexão.
 - Preencha os detalhes da conexão:
 - **MQTT Client Name:** test.mosquitto.org
 - **Protocol:** mqtt/tcp.
 - **Host:** test.mosquitto.org
 - Salve a configuração e conecte.

MQTTBox

MQTT CLIENT SETTINGS

MQTT Client Name: test.mosquitto.org

MQTT Client Id: dd0fcbd7-2130-432c-8f25-132c84f

Append timestamp to MQTT client id? ☒ Yes

Broker is MQTT v3.1.1 compliant? ☒ Yes

Protocol: mqtt / tcp

Host: test.mosquitto.org

Clean Session? ☒ Yes

Auto connect on app launch? ☒ Yes

Username: Username

Password: Password

Reschedule Pings? ☒ Yes

Queue outgoing QoS zero messages? ☒ Yes

Reconnect Period (milliseconds): 1000

Connect Timeout (milliseconds): 30000

KeepAlive (seconds): 10

Will - Topic: Will - Topic

Will - QoS: 0 - Almost Once

Will - Retain: ☐ No

Will - Payload:

Save Delete

3. Publicar e Assinar Tópicos:

- **Publicar:** Vá para a seção de publicação e insira os tópicos ‘umidade’ e ‘estado_irrigacao’. Clique em "Publicar".

test.mosquitto.org - mqtt://test.mosquitto.org

Topic to publish: umidade

QoS: 1 - Atleast Once

Retain ☒

Payload Type: Strings / JSON / XML / Characters

e.g: {hello:"world"}

Payload:

Publish

Topic to publish: estado_irrigacao

QoS: 1 - Atleast Once

Retain ☒

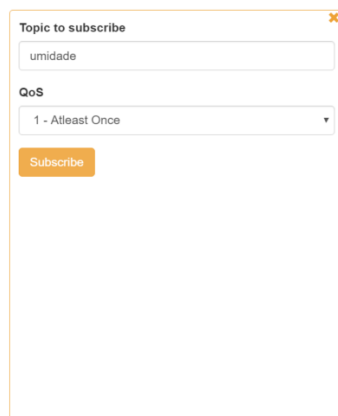
Payload Type: Strings / JSON / XML / Characters

e.g: {hello:"world"}

Payload:

Publish

- **Assinar:** Vá para a seção de assinatura, adicione tópicos ‘umidade’ e ‘estado_irrigacao’ e clique em "Assinar". As mensagens recebidas serão exibidas nesses tópicos.

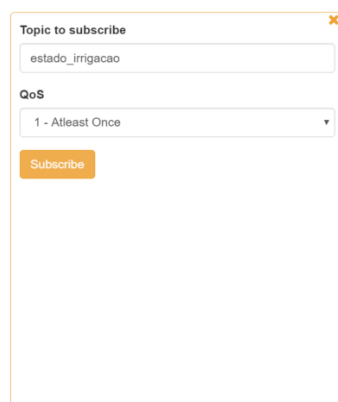


Topic to subscribe

QoS

1 - Atleast Once

Subscribe



Topic to subscribe

QoS

1 - Atleast Once

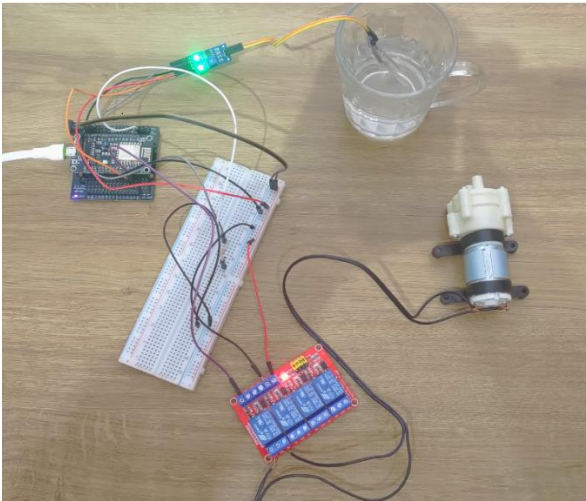
Subscribe

É importante pontuar que usar um broker público como o **test.mosquitto.org** é conveniente para testes e desenvolvimento, mas para aplicações em produção, é recomendável usar um broker privado ou um serviço de broker MQTT gerenciado para garantir segurança e confiabilidade. Além disso, certifique-se de gerenciar adequadamente as credenciais e a segurança da rede Wi-Fi ao desenvolver aplicações IoT.

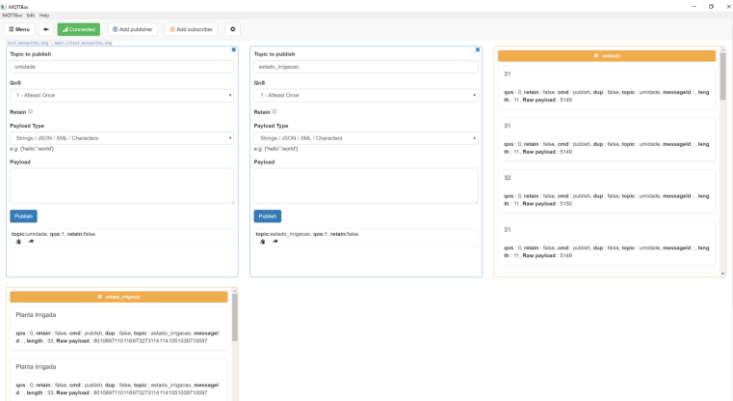
4. Resultados

O projeto final consiste em duas partes: hardware (o protótipo) e software. Será apresentado aqui o protótipo final montado e as funcionalidades obtidas através da sua integração com o software. Entre essas funcionalidades estão a capacidade de receber informações periódicas sobre a umidade atual e o estado do sistema de irrigação, podendo visualizá-las por meio de publicações no broker Mosquitto. Foi incluído também uma tabela com gráficos dos tempos de reação do sistema, link com o vídeo com a demonstração do funcionamento do protótipo e as publicações em tempo real das leituras do sensor de umidade e o estado da irrigação.

4.1 Protótipo



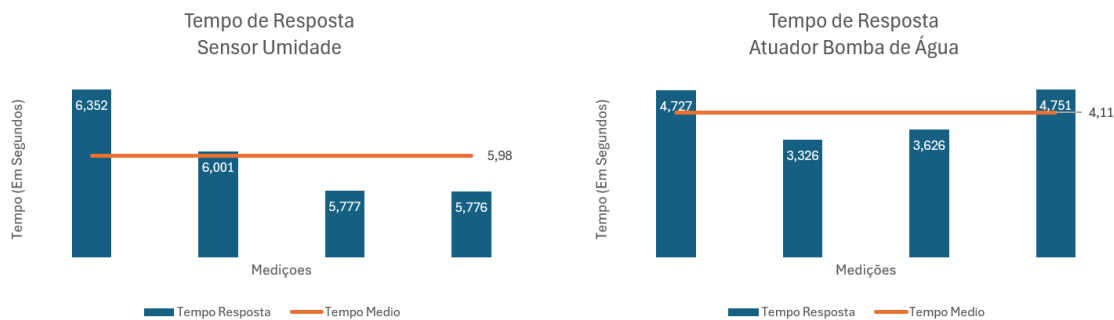
4.2 Publicação Broker Mosquito



4.3 Medição do Tempo de Resposta

Núm Medida	Elemento	Tempo Resposta	Tempo Medio
1	Sensor Umidade	6,352	5,98
2	Sensor Umidade	6,001	5,98
3	Sensor Umidade	5,777	5,98
4	Sensor Umidade	5,776	5,98

Núm Medida	Elemento	Tempo Resposta	Tempo Medio
1	Atuador Bomba Água	4,727	4,11
2	Atuador Bomba Água	3,326	4,11
3	Atuador Bomba Água	3,626	4,11
4	Atuador Bomba Água	4,751	4,11



4.4 Links

Link Repositório GitHub: <https://github.com/agnaldopires/MackIoT>

Link Vídeo do Sistema Funcionando: <https://youtu.be/CBJGhVwd3el>

5. Conclusões

Implementação do Sistema de Rega Automatizado

A implementação de um sistema de rega automatizado exigiu conhecimentos em diversas áreas, incluindo sistemas embarcados, atuadores e programação. Apesar de ser conceitualmente simples, o projeto apresentou desafios significativos, especialmente na integração entre o sistema embarcado e a publicação das leituras em um broker na nuvem, além da necessidade de aprendizado de um novo microprocessador.

Resultados

O resultado final foi um produto funcional e intuitivo. Todas as partes do sistema funcionam de forma independente, mas podem se comunicar entre si, proporcionando ao usuário uma aplicação simples e informativa. O sistema permite monitorar a umidade do solo e controlar a bomba de água de maneira eficiente, facilitando a vida de quem possui plantas, mas não tem tempo para monitorá-las constantemente.

Potencial para Melhorias

Existem várias melhorias possíveis, como a adição de sensores de fluxo e nível de água, a substituição do sensor de umidade por um mais resistente à corrosão, a personalização e o download dos dados coletados pelos usuários, além da criação de um dashboard para visualização gráfica das publicações.

Para melhorar a escalabilidade, futuras prototipagens podem incluir um relé com vários canais para controlar mais bombas de água e métodos de multiplexação para ler vários

sensores de umidade do solo com um único microprocessador. Isolar o circuito eletrônico da água com um case apropriado também seria uma medida importante para aumentar a segurança do sistema.

Respostas às Questões

i) Os objetivos propostos foram alcançados?

Sim, os objetivos propostos foram alcançados. O sistema é capaz de monitorar a umidade do solo e controlar a bomba de água de maneira automatizada e eficiente.

ii) Quais são os principais problemas enfrentados e como foram resolvidos?

Os principais problemas enfrentados incluíram a integração do sistema embarcado com o broker na nuvem e o aprendizado de um novo microprocessador. Esses desafios foram superados através de pesquisa, testes e iterativas melhorias no design e implementação do sistema.

iii) Quais são as vantagens e desvantagens do projeto?

- **Vantagens:**

- Automatiza a rega, economizando tempo para os usuários.
- Fornece monitoramento em tempo real da umidade do solo.
- Sistema modular que permite fácil expansão e personalização.

- **Desvantagens:**

- A integração inicial com o broker na nuvem pode ser complexa.
- O sensor de umidade pode ser suscetível à corrosão e desgaste ao longo do tempo.

iv) O que deveria/poderia ser feito para melhorar o projeto?

Para melhorar o projeto, poderiam ser implementadas as seguintes melhorias:

- Adição de sensores de fluxo e nível de água para monitoramento mais abrangente.
- Substituição do sensor de umidade por um modelo mais resistente à corrosão.
- Desenvolvimento de um dashboard para visualização gráfica dos dados coletados.
- Inclusão de um relé com vários canais para controlar múltiplas bombas de água.
- Uso de métodos de multiplexação para ler vários sensores de umidade com um único microprocessador.
- Isolamento do circuito eletrônico da água com um case apropriado para aumentar a segurança do sistema.

5. Referências

1. Smith, J., et al. (2018). Título do estudo. Revista de Agricultura Sustentável, 10(2), 123-135.
2. PANDEMIA e isolamento aumentam procura por cultivo de plantas em casa.
(<https://agenciabrasil.ebc.com.br/geral/noticia/2021-04/pandemia-e-isolamento-aumentam-procura-por-cultivo-de-plantas-em-casa>).
3. GITHUB oficial da NodeMCU.
(<https://github.com/nodemcu/nodemcu-devkit-v1.0>).