Дипломная работа

GeekBrains

Разработка и внедрение системы орошения почвы на дачном участке для ухода за растениями в теплице с возможностью автоматического и ручного режимов полива. Удаленное управление системой орошения из любой точки мира.

Разработчик — Инженер умных устройств
Могилат Сергей Андреевич

Санкт-Петербург
2024

Оглавление

Оглавление ... 1
Введение ... 2
Тема проекта ... 2
Цели проекта ... 2
Обоснование темы проекта ... 2
Проблематика ... 3
Задачи ... 3
Инструменты и Технологии ... 3
Команда ... 4
Глава 1. Техническое обоснование проекта ... 5
1.1 Краткое описание проекта ... 5
1.2 Значимость систем автоматизированного полива в современных условиях. ... 5
1.3 Анализ существующих технологий ... 7
1.3.1 Ручные системы полива ... 7
1.3.2 Капельный полив ... 8
1.3.3 Подповерхностный полив ... 8
1.3.4 Системы микроклимата ... 9
1.3.5 Автоматизированные системы полива ... 9
1.3.6 Сравнительный анализ технологий ... 10
1.3.7 Заключение ... 10
1.4 Выбор технической части проекта ... 10
1.4.1 Микроконтроллер ESP8266. ... 13
1.4.2 Датчик температуры и влажности воздуха DHT11. ... 15
1.4.3 Датчик влажности почвы YL-38 ... 19
1.5 Инструменты и платформы разработки ... 22
1.5.1 Среда разработки Arduino IDE ... 22
1.5.2 Протокол MQTT ... 23
1.5.3 Node-RED ... 24
1.5.4 InfluxDB ... 25
1.5.5 Grafana ... 26
1.5.7 Заключение ... 27
Глава 2. Проектирование и разработка системы ... 28
2.1 Архитектура и схема взаимодействия компонентов системы ... 28
2.2 Сборка устройства ... 30
2.3 Программирование системы ... 34
2.4 Тестирование ... 38
Заключение ... 42
Список используемой литературы ... 43

Введение

Проект заключается в разработке и внедрении системы автоматического и ручного увлажнения почвы на дачном участке для ухода за растениями, находящимися в теплице, с возможностью удаленного управления системой орошения из любой точки мира.

Проект основан на использовании микроконтроллера ESP8266, к которому подключены датчики влажности почвы и температуры и влажности воздуха DHT11. Эти датчики обеспечивают сбор данных, используемых для автоматического управления реле водяной помпы для полива растений. Вся собранная информация сохраняется и обрабатывается в базе данных InfluxDB, развернутой на удаленном сервере.

Тема проекта

Разработка и внедрение системы орошения почвы на дачном участке для ухода за растениями в теплице с возможностью автоматического и ручного режимов полива. Удаленное управление системой орошения из любой точки мира.

Цели проекта

Целью проекта является создание автономной системы, которая будет регулировать уровень влажности почвы, оптимизируя полив растений в теплице в соответствии с требуемыми условиями.

Обоснование темы проекта

Разработка данного проекта обоснована необходимостью создания современной автоматизированной системы полива, которая обеспечивает сбережение ресурсов посредством интеллектуального расхода воды, а также предоставляет возможность удаленного мониторинга и управления.

Проблематика

Проект будет направлен на решение проблемы необходимости оптимизации ухода за растениями в теплице в условиях сильно ограниченных ресурсов и удаленной локации.

В собственности имеется дачный участок, на котором расположены тепличные сооружения. Как известно, растения в теплице требуют тщательного ухода и регулярного полива. В связи с тем, что постоянно находиться в загородном доме и обеспечивать контроль за ростом растений не представляется возможным, возникла необходимость создания автоматизированной системы полива, которая обеспечит бесперебойную подачу воды в зависимости от внешних условий, когда это необходимо и без участия человека.

Задачи

1. Изучить рынок необходимых для проекта электронных компонентов и закупить их.
2. Написать код для реализации проекта.
3. Собрать устройство на основе микроконтроллера ESP8266.
4. Развернуть удаленный сервер для передачи и хранения информации с устройства.
5. Протестировать и настроить устройство на стенде.
6. Запустить устройство в работу в реальных условиях для выявления и устранения недоработок.
7. Убедиться, что работа устройства полностью соответствует целям и проблематике проекта.

Инструменты и Технологии

Реализация проекта осуществляется при помощи микроконтроллера ESP8266, датчиков влажности почвы и температуры и влажности воздуха DHT11, а также базы данных InfluxDB для хранения и обработки собранных

данных. Технологии, использованные в проекте, включают разработку ПО и работу с базами данных для обработки данных датчиков и удаленного управления системой орошения.
Используемые инструменты: Git, MQTT, Grafana, Node-Red, InfluxDB, ArduinoIDE, VSCode.

Команда

Реализация проекта была осуществлена индивидуально, включая настройку микроконтроллера, написание кода, тестирование и запуск рабочего образца.

Глава 1. Техническое обоснование проекта

1.1 Краткое описание проекта

На моем дачном участке, где находятся тепличные сооружения, требуется установить систему автоматического полива. Обычно для растений в теплице необходим тщательный уход, включая регулярный полив. Однако, поскольку постоянное присутствие на даче для обеспечения контроля за поливом невозможно, возникла необходимость в создании автоматизированной системы полива. Эта система будет обеспечивать непрерывное снабжение водой в соответствии с внешними условиями, даже в отсутствие человека.

1.2 Значимость систем автоматизированного полива в современных условиях.

Помимо моей «локальной» задачи, значимость систем автоматического полива ежегодно растет во всем мире. Автоматизированные системы полива становятся все более актуальными в условиях нарастающего дефицита пресной воды, изменения климата и роста потребностей в сельском хозяйстве. Такие системы обеспечивают эффективное использование ресурсов и оптимизацию ухода за растениями.

Ниже представлены основные причины актуальности подобных систем:

1. Экономия времени и ресурсов.

Автоматизированный полив значительно снижает потребность в ручном уходе за растениями. Это особенно важно, как и для владельцев дач и теплиц, которые часто не могут проводить много времени на своем участке, так и для больших промышленных хозяйств.

Системы, использующие датчики влажности, способны точно определять, когда и сколько воды нужно подать, что позволяет исключить перерасход воды и, соответственно, сохранить водные ресурсы.

2. Оптимизация условий для роста растений.

Автоматизированные системы могут поддерживать оптимальный уровень влажности почвы, что способствует здоровому росту и развитию растений.

Автоматизированные системы способны адаптироваться к изменениям в окружающей среде, что обеспечивает менее стрессовые условия для растений в условиях с жарким климатом или редкими дождями.

3. Удаленное управление и мониторинг.

Возможность удаленного контроля за системой полива через мобильные приложения и веб-интерфейсы позволяет владельцам управлять поливом из любой точки мира.

Удаленный мониторинг позволяет не только контролировать уровень влажности, но и следить за состоянием растений в реальном времени, что улучшает общую эффективность ухода.

4. Устойчивое развитие и экология.

Автоматизированные системы помогают снизить количество избыточного полива, что ведет к меньшему стоку и эрозии почвы, а также уменьшает риск загрязнения подземных вод удобрениями и пестицидами.

В условиях изменения климата, когда традиционные методы полива могут оказаться неэффективными, инновационные технологии обеспечивают устойчивое развитие агрономии.

5. Инновационные подходы.

Автоматизированные системы могут интегрироваться с другими технологиями, такими как дроны для мониторинга состояния растений, системы управления микроклиматом и т.д. Это создает целостную экосистему управления ухом за растениями.

Внедрение автоматизированных систем полива в современные условия становится не только технологически необходимым, но и экономически целесообразным. Они обеспечивают эффективное распределение ресурсов, оптимизацию ухода за растениями и отвечают вызовам современного сельского хозяйства и экологии. Технологический прогресс в этой области открывает новые горизонты для более эффективного и устойчивого ведения сельского хозяйства.

1.3 Анализ существующих технологий

В современном сельском хозяйстве и садоводстве существует множество технологий для полива растений. Они варьируются от традиционных методов до современных автоматизированных систем. Понимание каждой технологии помогает выбрать оптимальный подход для конкретных условий.

1.3.1 Ручные системы полива

Описание: Ручные системы полива включают шланги, лейки и поливальные устройства, которые требуют физического участия человека.

Преимущества:

- Низкая стоимость в оборудовании.
- Простота использования, не требующая специальных навыков.

Недостатки:

- Большие трудозатраты при частом поливе.
- Нет возможности контролировать количество подаваемой воды, что может привести к избыточному или недостаточному поливу.
- Человеческий фактор может влиять на регулярность полива.

1.3.2 Капельный полив

Описание: Эта система включает в себя использование трубок с небольшими отверстиями, через которые вода поступает непосредственно к корням растений.

Преимущества:

- Высокая эффективность использования воды, так как подача осуществляется прямо к корням.
- Уменьшение роста сорняков благодаря отсутствию влаги на поверхности почвы.

Недостатки:

- Более высокая стоимость установки и обслуживания по сравнению с ручными системами.
- Засорение капельниц органическими остатками и минеральными отложениями может повредить систему.

1.3.3 Подповерхностный полив

Описание: В этой системе трубки устанавливаются под поверхностью почвы, чтобы подавать воду непосредственно к корням.

Преимущества:

- Эффективно сохраняет влагу и минимизирует испарение.
- Предотвращает развитие грибковых заболеваний, связанных с избыточной влажностью верхнего слоя почвы.

Недостатки:

- Высокая стоимость установки и сложность в обслуживании.
- Необходимость предварительной планировки, что затрудняет изменение схемы полива.

1.3.4 Системы микроклимата

Описание: Это системы, которые регулируют полив в зависимости от метеорологических условий и состояния растений.

Преимущества:

- Отслеживание реального времени позволяет эффективно использовать ресурсы, сокращая потребление воды.
- Подходит для больших коммерческих хозяйств, где точность и эффективность критичны.

Недостатки:

- Высокая стоимость оборудования и настройки.
- Необходимость технического обслуживания и постоянного мониторинга.

1.3.5 Автоматизированные системы полива

Описание: Эти системы могут включать датчики влажности, таймеры и микроконтроллеры, которые автоматически контролируют полив на основе данных.

Преимущества:

- Значительное снижение трудозатрат благодаря автоматизации процессов.
- Возможность интеграции с другими системами (например, метеостанции, системы управления растениями).
- Доступ к удаленному управлению через интернет и мобильные приложения.

Недостатки:

- Начальные затраты на оборудование могут быть высокими.

- Зависимость от электроэнергии и надежности интернета, что может быть проблемой в удаленных районах.

1.3.6 Сравнительный анализ технологий

Эффективность: Автоматизированные и капельные системы обеспечивают наилучшее распределение воды, в отличие от ручных методов.

Стоимость: Ручные и капельные системы имеют меньшие начальные затраты, однако обслуживание и эксплуатационные расходы могут увеличиться для менее эффективных систем.

Трудозатраты: Ручные системы требуют максимум труда, в то время как автоматизированные решения значительно снижают нагрузку, но требуют первоначальных затрат на установку.

1.3.7 Заключение

Каждая из существующих технологий полива имеет свои преимущества и недостатки. Выбор оптимальной системы зависит от множества факторов, включая тип растений, климатические условия, доступные ресурсы и бюджет. Знание этих технологий помогает принять более обоснованное решение, что особенно важно в условиях современных вызовов, таких как нехватка водных ресурсов и изменение климата.

1.4 Выбор технической части проекта

Как я уже показал выше, на данный момент на рынке доступен широкий ассортимент устройств для автоматизированного орошения, начиная от простых механических до сложных «умных» промышленных систем. Однако, при рассмотрении этих вариантов для моей специфической задачи, я обнаружил, что они либо не обладают достаточной эффективностью, либо являются излишне дорогими. Именно поэтому я принял решение самостоятельно собрать мою систему.

После изучения рынка необходимых компонентов, за основу я выбрал микроконтроллер ESP8266 как наиболее подходящий для моего проекта. ESP8266 представляет собой отличный выбор благодаря тому, что совмещает в себе доступность, функциональность и надежность. В сравнении с аналогичными устройствами, ESP8266 отличается встроенным Wi-Fi модулем, высокой эффективностью, низкой стоимостью и широкой поддержкой сообщества разработчиков. Возможности программирования данного микроконтроллера, доступность ресурсов и стабильность работы делают его превосходным выбором для создания автоматизированной системы полива на дачном участке.

Для сбора данных об окружающей среде, важных для автоматизированной системы орошения, было принято решение включить в систему датчик температуры и влажности воздуха, а также датчик влажности почвы. Хотя для базового функционирования системы достаточно только датчика влажности почвы, в контексте создания полной системы контроля за условиями в теплице, данные о влажности и температуре воздуха предоставляют дополнительную информацию, которая может быть использована для более глубокого анализа условий роста растений и последующей модификации системы орошения. В качестве датчика контроля за состоянием воздуха был выбран простой, дешевый и надежный DHT11, а для анализа почвы - YL-38.

Система полива не может быть полноценной без водяной помпы, поэтому в качестве основного устройства для подачи воды был приобретен водяной насос, работающий на напряжении 5 Вольт. Для управления этим водяным насосом необходимо использовать реле, которое получает сигнал от микроконтроллера, и включает либо отключает подачу воды в систему полива. Один из оптимальных выборов для этой задачи – это одноканальный релейный модуль постоянного тока 5 Вольт с максимальным током нагрузки 10А. Это

решение обеспечивает необходимую функциональность и соответствует требованиям проекта для эффективной работы системы полива.

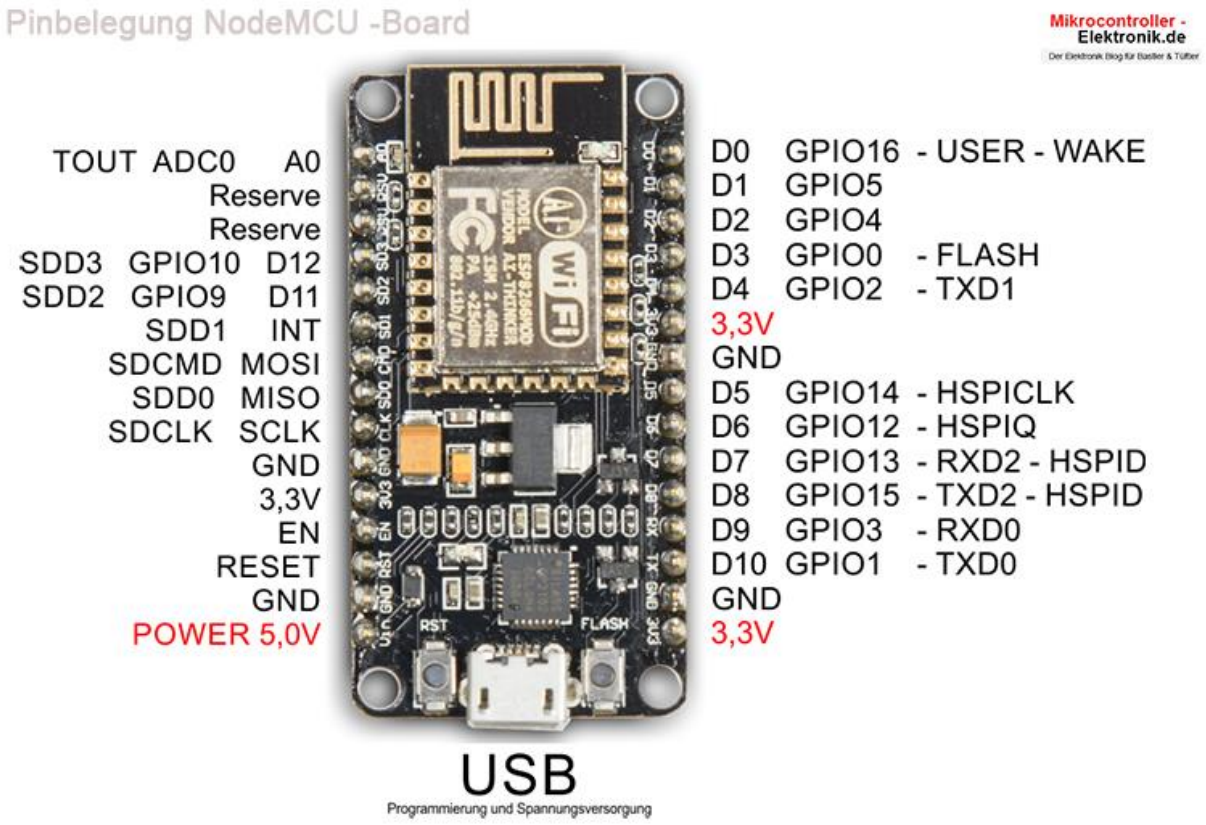
В качестве сервера для хранения и обработки данных с основной системы был выбран имеющийся у меня одноплатный компьютер Raspberry Pi 3. Выбор был обоснован тем, что устройство потребляет мало энергии, что существенно сэкономит затраты на электроэнергию при работе 24/7, а также отсутствием необходимости брать в аренду удаленный сервер.

Таким образом список компонентов системы будет выглядеть следующим образом:

- Контроллер Wi-Fi NodeMCU V3 на базе ESP8266;
- Датчик температуры и влажности воздуха DHT11;
- Датчик влажности почвы YL-38;
- Релейный модуль постоянного тока 5V 10A;
- Водяной насос 5V;
- Одноплатный компьютер Raspberry Pi 3.

Очевидно, что в моем проекте использована маломощная помпа, что достаточно для демонстрации работы в стендовых условиях. Однако на практике для реальной эксплуатации системы потребуется заменить насос на более мощный, работающий от обычной бытовой сети переменного тока напряжением 220 Вольт. Предложенная схема практически не требует дальнейших изменений, однако для корректной работы системы будет необходим дополнительный источник питания, а также замена реле и самого насоса. Далее рассмотрим составные части системы по пунктам.

1.4.1 Микроконтроллер ESP8266.



Контроллер Wi-Fi NodeMCU V3 на базе ESP8266

ESP8266 — это недорогой и мощный микроконтроллер, который стал популярным в разработке IoT-решений благодаря своей интеграции возможностей Wi-Fi. Он предоставляет разработчикам уникальную возможность соединять физические устройства с интернетом, что делает его идеальным выбором для автоматизации систем, таких как системы орошения.

Архитектура и технические характеристики:

В основе ESP8266 лежит 32-битный процессор Xtensa, работающий на частоте до 80 МГц, что обеспечивает быструю обработку данных. Оперативная и флэш-память: В большинстве моделей присутствует 80 КБ ОЗУ и 4 МБ или 16 МБ флэш-памяти для хранения прошивок и данных. ESP8266 поддерживает различные интерфейсы связи, в том числе:

- GPIO (общие входы/выходы): обычно до 17 линий ввода/вывода, которые можно использовать для подключения различных датчиков и исполнительных механизмов.

- I2C, SPI и UART: позволяют взаимодействовать с внешними устройствами и датчиками.
- PWM (широтно-импульсная модуляция): для управления аналоговыми устройствами, такими как насосы и датчики.

ESP8266 может подключаться к Wi-Fi сетям с помощью всего лишь нескольких строк кода, что делает его доступным даже для начинающих разработчиков. Встроенная поддержка TCP/IP стеков позволяет организовать передачу данных через HTTP, MQTT и другие протоколы, что идеально подходит для приложений IoT.

ESP8266 можно программировать с использованием Arduino IDE с помощью специальной библиотеки, что делает разработку еще более наглядной и доступной. Программирование на Arduino дает возможность использовать огромное количество готовых библиотек и примеров кода.

Преимущества использования ESP8266 в проекте:

- Низкая стоимость: Один из самых доступных микроконтроллеров на рынке, что позволяет существенно сократить бюджет проекта.
- Компактность: Небольшие размеры модуля позволяют легко интегрировать его в различные устройства.
- Гибкость: ESP8266 может использоваться в самых разных приложениях — от простых датчиков до сложных автоматизированных систем.

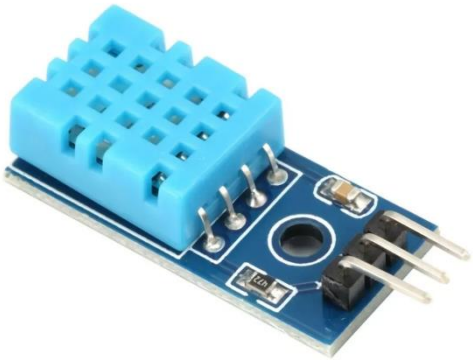
Ограничения ESP8266:

- Мощность: Микроконтроллер потребляет достаточно много энергии, особенно при активной передаче данных, что может быть критично для автономных систем, работающих от батарей.
- Ограничения по количеству соединений: ESP8266 ограничен по количеству одновременных соединений с клиентами и сервером, что может стать проблемой при массовом использовании.

Заключение.

ESP8266 — это мощный и доступный инструмент для разработки IoT-приложений, включая системы автоматического полива. Его возможности, такие как поддержка Wi-Fi, простота программирования и обширная экосистема библиотек и сообществ, делают его идеальным выбором для реализации инновационных решений в сфере сельского хозяйства и других областях.

1.4.2 Датчик температуры и влажности воздуха DHT11.



Датчик температуры и влажности воздуха DHT11.

DHT11 — это недорогой и простой в использовании цифровой датчик, который предназначен для измерения температуры и влажности воздуха. Он широко применяется в системах автоматизации, метеорологии, умных домах и других проектах IoT.

Основные характеристики:

- Измеряемые параметры: Температура и влажность.
- Диапазон измерения влажности: от 20% до 80% с точностью ±5% RH.
- Диапазон температур: от 0°C до 50°C с точностью ±2°C.
- Выходной сигнал: цифровой, на одном проводе (объединение сигналов для температуры и влажности).
- Питание: обычно 3.5V до 5.5V, что делает его совместимым с большинством микроконтроллеров.

- Скорость обмена данными: до 1 раза в секунду.

Принцип работы.

DHT11 использует комбинацию полупроводниковой технологии и термисторов для измерения температуры и влажности. Внутри датчика расположены специальные сенсоры, которые реагируют на изменения температуры и влажности воздуха.

Сенсор влажности: работает на основе изменения сопротивления полимерного материала, который изменяет свои свойства в зависимости от уровня влажности.

Температурный сенсор: использует термистор, который изменяет свое сопротивление с изменением температуры, что позволяет определить ее уровень.

Подключение DHT11 к микроконтроллеру.

Для подключения датчика к микроконтроллеру (например, ESP8266 или Arduino) нужно следовать нескольким основным шагам:

- VCC: Подключается к источнику питания (обычно 5V).
- GND: Подключается к земле.
- DATA: Подключается к любому цифровому выводу микроконтроллера (необходим резистор подтяжки).
- Использование библиотеки для обработки данных: Библиотеки, такие как "DHT sensor library" для Arduino, упрощают работу с DHT11 и позволяют легко считывать данные о температуре и влажности.

Программирование.

Пример кода для Arduino, который показывает, как считывать данные с DHT11:

```
#include "DHT.h"

#define DHTPIN 2          // Пин, к которому подключён датчик DHT11
#define DHTTYPE DHT11     // Определяем тип датчика

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}
void loop() {
  delay(2000);  // Задержка между считываниями

  float h = dht.readHumidity();  // Чтение температуры и влажности
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t)) {
    Serial.println("Ошибка считывания с DHT11!");  // Проверка на ошибки
    return;
  }

  Serial.print("Влажность: "); // Вывод результатов
  Serial.print(h);
  Serial.print(" %\t");
  Serial.print("Температура: ");
  Serial.print(t);
  Serial.println(" *C");
}

```

Преимущества и недостатки.

Преимущества:

- Стоимость: DHT11 — один из самых доступных сенсоров на рынке.
- Простота использования: Легко подключается и программируется.
- Компактность: Небольшие габариты позволяют интегрировать его в разные конструкции.

Недостатки:

- Точность: По сравнению с более дорогими аналогами, такими как DHT22, DHT11 менее точен, особенно в крайних диапазонах.
- Диапазон измерения: Ограниченный диапазон температур и влажности по сравнению с другими датчиками.
- Скорость реакции: Может быть медленнее, чем более сложные датчики.

Применение.

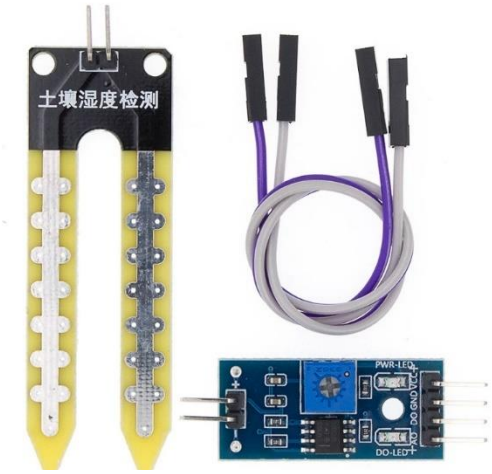
DHT11 находит применение в различных областях:

- Системы автоматического полива: для мониторинга влажности воздуха и определения необходимости полива растений.
- Умные дома: для управления климатом в помещениях.
- Метеорология: в метеостанциях для сбора данных о погоде.
- Проекты для обучения: благодаря простоте использования идеален для начинающих.

Заключение.

DHT11 — это полезный и доступный сенсор для измерения температуры и влажности воздуха. Он отлично подходит для небольших проектов IoT и прекрасно работает в сочетании с популярными микроконтроллерами. Несмотря на некоторые ограничения, такие как точность и диапазон, его достоинства делают его идеальным выбором для простых устройств.

1.4.3 Датчик влажности почвы YL-38



Датчик влажности почвы YL-38

YL-38 — это экономичный датчик, используемый для определения уровня влажности в почве. Он часто применяется в системах полива, автоматизации садов и теплиц, а также в проектах для умных домов и IoT.

Основные характеристики

- Рабочий диапазон влажности: Обычно от 0% до 100% (влажность почвы).
- Рабочее напряжение: Обычно от 3.3 до 5V DC, что делает его совместимым с различными микроконтроллерами.
- Выходной сигнал: Датчик выдает аналоговый сигнал, отражающий уровень влажности.

Принцип работы

YL-38 работает на основе изменения электрического сопротивления в зависимости от уровня влажности. Это достигается с помощью металлических электродов, которые помещаются в почву:

Электроды повреждаются коррозией со временем, что может влиять на точность измерений. Они создают электрическую цепь, сопротивление которой меняется в зависимости от влажности почвы (чем больше влажность, тем меньше сопротивление).

Чем выше уровень влажности, тем выше выходной сигнал, что позволяет легко интегрировать датчик в систему сбора данных.

Подключение YL-38 к микроконтроллеру

Подключение датчика достаточно простое и требует трех основных соединений:

- VCC: Подключается к 3.3V или 5V на микроконтроллере.
- GND: Подключается к земле (GND).
- A0: Подключается к аналоговому входу на микроконтроллере для считывания значения влажности. Также существует цифровой вывод (D0), который можно использовать для порогового срабатывания.

Программирование

Пример кода для Arduino, использующего YL-38:

```
const int sensorPin = A0;  // Подключаем выход A0 к аналоговому пину Arduino
int sensorValue;

void setup() {
  Serial.begin(9600);  // Запуск последовательного соединения
}

void loop() {
  sensorValue = analogRead(sensorPin);  // Чтение значения с датчика
  Serial.print("Влажность почвы: ");
  Serial.println(sensorValue);  // Вывод значения на последовательный монитор
  delay(1000);  // Пауза в 1 секунду
}
```

Преимущества и недостатки

Преимущества:

- Низкая стоимость: YL-38 является одним из наиболее доступных датчиков на рынке, что делает его идеальным для образовательных и хоббийных проектов.
- Простота использования: Легко подключается и интегрируется с микроконтроллерами.
- Аналоговый выход: Позволяет получать непрерывные данные о влажности.

Недостатки:

- Коррозия: Металлические электроды могут корродировать при длительном использовании в почве, что снижает точность и срок службы датчика.
- Чувствительность к температуре: Может давать неточные показания при изменении температуры окружающей среды.
- Необходимость калибровки: Для обеспечения точности измерений может потребоваться периодическая калибровка.

Применение

YL-38 находит применение в следующих областях:

Системы автоматического полива: Для определения уровня влажности и автоматического включения или выключения полива.

Сади и огороды: Для мониторинга условий роста растений.

Умные дома: Интеграция в системы управления для создания комфортной домашней среды.

Образовательные проекты: Широко используется в образовательных целях для изучения принципов работы датчиков и автоматизации.

Заключение

YL-38 — это простой и доступный датчик влажности почвы, который может быть использован в различных проектах, связанных с агрономией и автоматизацией. Несмотря на некоторые недостатки, такие как чувствительность к коррозии и необходимость калибровки, его преимущества делают его отличным выбором для многих проектов, требующих мониторинга влажности почвы.

1.5 Инструменты и платформы разработки

1.5.1 Среда разработки Arduino IDE

Arduino IDE (Integrated Development Environment) — это бесплатная и открытая среда разработки, предназначенная для программирования плат Arduino и совместимых микроконтроллеров.

Интерфейс интуитивно понятен и доступен даже для начинающих разработчиков. Создание и загрузка проектов на плату осуществляется в несколько простых шагов. Для программирования используется упрощенный язык на основе C/C++. Arduino IDE содержит множество встроенных библиотек для работы с различными компонентами. Возможность подключения внешних библиотек значительно ускоряет разработку по сравнению с написанием кода с

нуля. Монитор последовательного порта позволяет пользователям видеть вывод данных от платы, что удобно для отладки. Arduino IDE работает на Windows, macOS и Linux, что делает ее доступной для широкого круга пользователей.

Преимущества Arduino IDE:

- Обширное сообщество и множество ресурсов для обучения, включая учебники, видео и проекты.
- Простота работы с различными датчиками и периферийными устройствами благодаря большому количеству библиотек.
- Быстрый доступ к функциям и библиотекам, что существенно ускоряет процесс разработки.

1.5.2 Протокол MQTT

MQTT (Message Queuing Telemetry Transport) — это легковесный протокол обмена сообщениями, разработанный для передачи данных между устройствами в условиях ограниченной пропускной способности сети и высоких задержек.

Протокол включает минимальный набор заголовков, что делает его идеальным для устройств с ограниченными ресурсами. MQTT работает на основе TCP, что обеспечивает надежный обмен сообщениями. Модель «публикация/подписка» позволяет устройствам отправлять данные (публикация) и получать их (подписка) без необходимости прямого подключения. Протокол оптимизирован для минимизации времени доставки сообщений, что делает его идеальным для IoT-приложений.

Где используется MQTT:

- В системах домашней автоматизации для управления освещением, безопасностью и другими системами.

- В IoT-решениях, где необходимо быстро передавать данные между множеством устройств.
- Например, на платформах мониторинга окружающей среды, где датчики отправляют данные о состоянии атмосферы.

Преимущества MQTT:

- Эффективное использование полосы пропускания, что особенно важно для систем с низкой мощностью и ресурсов.
- Надежная гарантированная доставка сообщений с возможностью настройки различных уровней QoS (Quality of Service).
- Подходит для работы в реальном времени, обеспечивая мгновенные обновления состояния устройств.

1.5.3 Node-RED

Node-RED — это визуальный инструмент разработки, использующий технику «перетаскивания» для создания приложений IoT и веб-приложений. Он построен на базе JavaScript и работает на платформе Node.js.

Графический интерфейс позволяет пользователям создавать потоки данных, перетаскивая различные узлы (nodes) и соединяя их между собой. Node-RED поддерживает множество протоколов, включая HTTP, MQTT, WebSocket и другие, что облегчает интеграцию с различными устройствами и сервисами. Позволяет добавлять новые узлы и функциональность через библиотеку пакетов, увеличивая возможности платформы.

Где используется Node-RED:

- Для разработки сложных IoT-решений, включая автоматизацию домашнего хозяйства, системы мониторинга и управления данными.

- В интеграциях с различными веб-сервисами, например, для отправки уведомлений в мессенджеры или сохранения данных в облачные хранилища.

Преимущества Node-RED:

- Удобный и интуитивно понятный интерфейс, что делает его доступным для пользователей с различным уровнем подготовки.
- Поддержка быстрого прототипирования, что позволяет разработчикам тестировать идеи и интеграции в считанные минуты.
- Широкое сообщество, которое развивает проект и создает новые узлы с уникальной функциональностью.

1.5.4 InfluxDB

InfluxDB — это высокопроизводительная база данных временных рядов, специально разработанная для обработки больших объемов данных, поступающих из различных источников в реальном времени.

InfluxDB оптимизирована для временных рядов. Специально разработана для хранения и анализа данных с привязкой ко времени, что делает ее идеальной для IoT-приложений. Способна обрабатывать миллионы записей в секунду, что позволяет строить масштабируемые решения. Входит в состав TICK Stack (Telegraf, InfluxDB, Chronograf, Kapacitor), что предоставляет мощные инструменты для мониторинга и анализа.

Где используется InfluxDB:

- В системах мониторинга для хранения метрик производительности, таких как информация о состоянии серверов и сетей.
- Для хранения данных от IoT-устройств, где важна временная составляющая (например, температура, влажность, давление).

Преимущества InfluxDB:

- Простота работы с данными с помощью собственного языка запросов (InfluxQL).
- Высокая степень сжатия данных, что позволяет экономить место на диске и распределять нагрузку.
- Хорошая интеграция с другими инструментами анализа, такими как Grafana.

1.5.5 Grafana

Grafana — это платформа для визуализации и анализа данных, которая позволяет создавать интерактивные графики и панели мониторинга, используя данные из различных источников, включая InfluxDB, Prometheus и другие базы данных.

Интерактивные панели мониторинга позволяют пользователям настраивать панели с графиками, таблицами и другими визуальными элементами для представления данных. Grafana может интегрироваться с различными источниками данных, что делает её универсальным инструментом для мониторинга разных параметров. Grafana поддерживает систему предупреждений, которая может уведомлять пользователей по электронной почте или SMS при выполнении заданных условий.

Где используется Grafana:

- В системах мониторинга данных от IoT-устройств для визуализации состояния системы в реальном времени.
- Для отображения метрик производительности серверов и приложений, что помогает в принятии решений о производительности и масштабировании.

Преимущества Grafana:

- Простота настройки и использования, что позволяет пользователям быстро интегрировать её в свои проекты.
- Возможность создания сложных графиков и схематичных представлений данных для достижения более глубокого понимания.
- Обширная библиотека плагинов, что позволяет расширять функциональность Grafana.

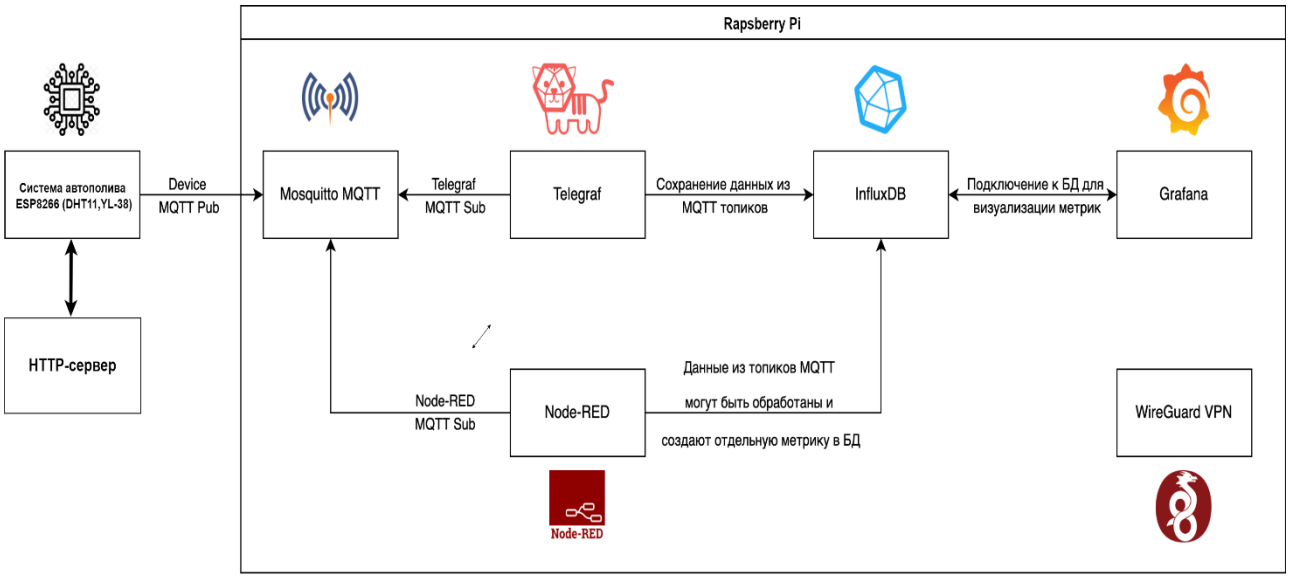
1.5.7 Заключение

Таким образом, различные инструменты и платформы разработки, такие как Arduino IDE, MQTT, Node-RED, InfluxDB и Grafana, обеспечивают мощные и гибкие средства для создания, развертывания и мониторинга IoT-приложений. Использование этих технологий позволяет разработчикам быстро создавать эффективные решения, интегрируя устройства и обеспечивая надежный анализ данных в реальном времени. Именно поэтому данные сервисы стали основными для реализации моего проекта.

Глава 2. Проектирование и разработка системы

2.1 Архитектура и схема взаимодействия компонентов системы

При создании подобных проектов важно заранее продумать, как будут взаимодействовать друг с другом все компоненты системы. Для того, чтобы наглядно сформировать принцип работы устройства и то, какие инструменты и протоколы связи будут задействованы, я использовал сервис Draw.io. Ниже представлена схема, которая поможет понять, как устройство получает и передает данные, для их последующего хранения и анализа.



Устройство на базе ESP8266 получает информацию о внешней среде с датчиков DHT11 и YL-38. Управляет подачей воды, анализируя информацию, полученную с датчиков. Передает данные на локальный HTTP сервер, который в свою очередь позволяет не только просматривать информацию об устройстве и данные с датчиков, но и управлять устройством.

В схеме присутствует одноплатный компьютер Rapsberry Pi 3, который выступает в качестве сервера для работы с данными. На RPi3 развернуты все необходимые сервисы, такие как Telegraf, Node-RED, InfluxDB и Grafana.

Через протокол MQTT данные об окружающей среде отправляются в инструмент разработки Node-RED, а далее записываются в базу данных InfluxDB и визуализируются в Grafana.

Для доступа к системе из внешней сети на роутере развернут VPN Wireguard. Подключившись к VPN из любой внешней сети, можно получить доступ к устройствам, находящимся в локальной сети. Таким образом, можно управлять системой автополива почвы из любой точки мира.

Схема взаимодействия компонентов выглядит следующим образом:

- ESP8266 собирает данные с датчиков DHT11 и YL-38.
- Данные передаются на локальный HTTP сервер.
- С помощью протокола MQTT, данные отправляются на сервер, который развернут на Rapsberry Pi, сначала в Node-RED.
- Node-RED обрабатывает данные и направляет их в InfluxDB.
- InfluxDB сохраняет данные, что позволяет осуществлять долгосрочный мониторинг.
- Grafana визуализирует данные, предоставляя удобные графические интерфейсы для анализа состояния окружающей среды и управления устройством.

При проектировании IoT-систем важно не только подобрать правильные компоненты, но и тщательно продумать их взаимодействие. Использование сервисов, таких как Draw.io, позволяет визуализировать архитектуру системы, а применение протоколов и инструментов, таких как MQTT, Node-RED, InfluxDB и Grafana, обеспечивает эффективное управление данными и их анализ в реальном времени. Это создает оптимальные условия для достижения поставленных задач и внедрения высококачественных решений в области автоматизации.

2.2 Сборка устройства

Прежде чем приступить к сборке системы, было принято решение о важности предварительной визуализации, что позволит наглядно представить электрические схемы и компоненты. Системы визуализации помогают не только упростить процесс проектирования, но и снизить вероятность ошибок при реальной сборке устройства.

При выборе подходящего инструмента для визуализации я изучил несколько популярных сервисов, среди которых были:

Tinkercad. Удобный онлайн-сервис для проектирования электрических схем и 3D-моделирования. Однако его функциональность в области схем и компонентов была ограничена по сравнению с другими платформами.

EasyEDA. Это мощный инструмент, который интегрирует функционал схем, печатных плат и симуляций. Он позволяет создавать сложные схемы и PCB, однако интерфейс оказался не столь интуитивно понятным для моих целей.

CircuitLab. Хороший инструмент для составления электрических схем с возможностью симуляции, но в нем отсутствовал богатый библиотечный набор компонентов, что ограничивало использование.

Fritzing. Это средство визуализации, которое выделялось среди других благодаря своему интуитивному интерфейсу и обширной библиотеке компонентов. Оно позволяет создавать как прототипы на макетной плате, так и полные схемы с деталями, что крайне полезно для работы с микроконтроллерами и различными датчиками.

После тщательного анализа я выбрал Fritzing по нескольким причинам. Fritzing имеет очень дружелюбный и интуитивно понятный пользовательский интерфейс. Даже для новичков работа с программой не вызывает затруднений, так как все необходимые инструменты расположены на видном месте и легко

доступны. Программа предлагает обширную библиотеку компонентов – от распространенных микроконтроллеров, таких как Arduino и ESP8266, до различных датчиков, резисторов и других электронных элементов. Это значительно упрощает процесс сборки, ведь я могу быстро находить и добавлять нужные элементы. Fritzing позволяет экспортировать схемы в различные форматы, включая PDF и изображения, что удобно для документирования проекта и его дальнейшей демонстрации. Существующее сообщество Fritzing активно делится схемами, что дает возможность получить вдохновение и готовые решения для различных проектов. Платформа предлагает доступ к качественной документации и множеству примеров, что значительно упрощает процесс обучения и погружения в проект.

Работа с Fritzing началась с создания макета системы. Я разработал несколько вариантов схем, добавляя компоненты и организуя их на макетной плате, чтобы наглядно отразить все соединения.

Я начал с добавления микроконтроллера ESP8266 на макетную плату, а затем добавил датчики DHT11 и YL-38, а также управляющие элементы, такие как реле для подачи воды.

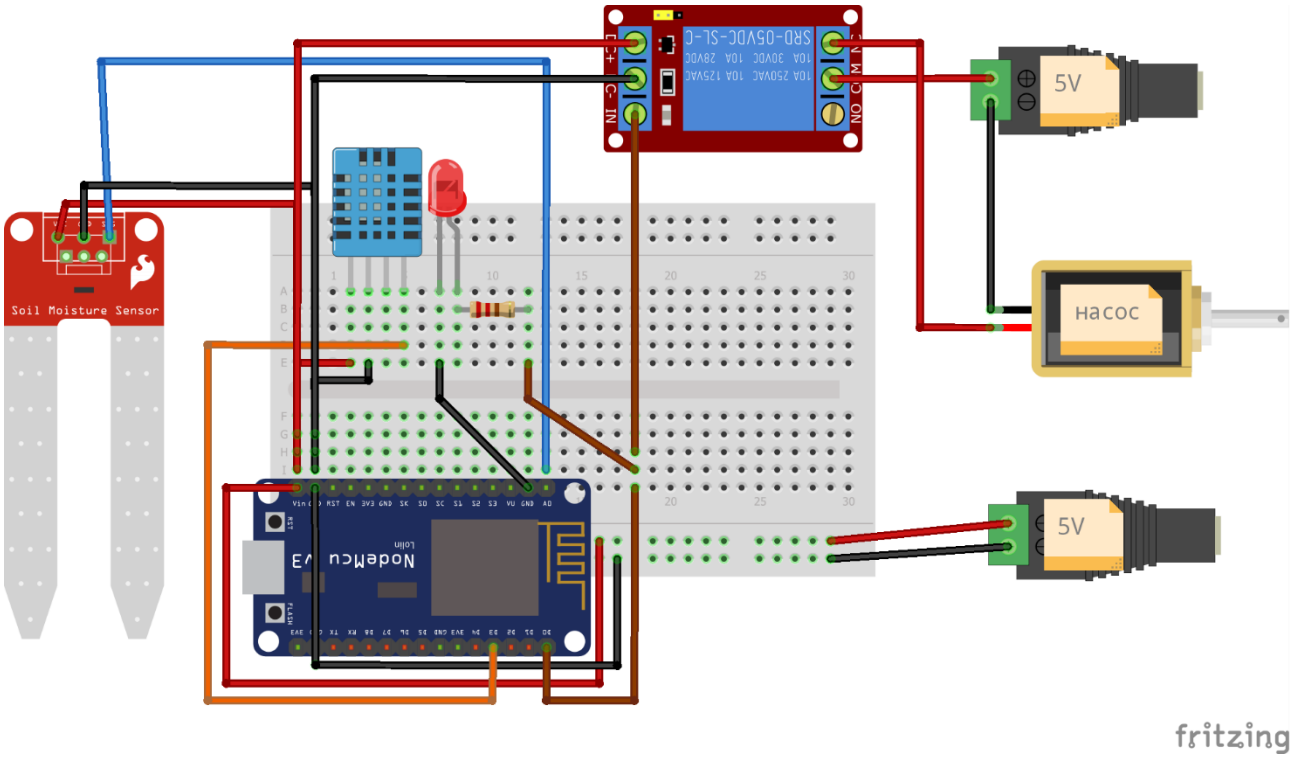
Следующим этапом стало создание соединений между компонентами. Я использовал цветные проводники для обозначения различных сигналов (например, питание, земля, данные), что облегчило восприятие схемы.

После завершения схемы я детально проверил все соединения и компоненты на наличие ошибок, чтобы убедиться, что схема будет правильно функционировать при сборке.

Разработанная схема была экспортирована в формат .png для дальнейшего использования. Это не только поможет в процессе сборки, но и послужит основой для документации проекта.

Выбор Fritzing в качестве инструмента для визуализации оказался удачным решением, которое значительно упростило процесс проектирования системы. Благодаря интуитивно понятному интерфейсу, обширной библиотеке компонентов и возможным вариантам экспорта, я смог не только эффективно спланировать свою систему, но и подготовить качественную документацию для дальнейшего производства и анализа.

Ниже представлена схема устройства, разработанная в Fritzing.

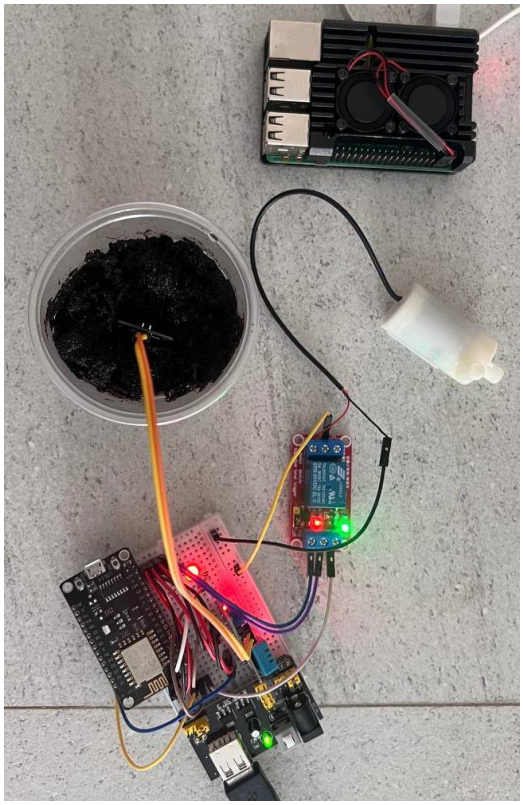


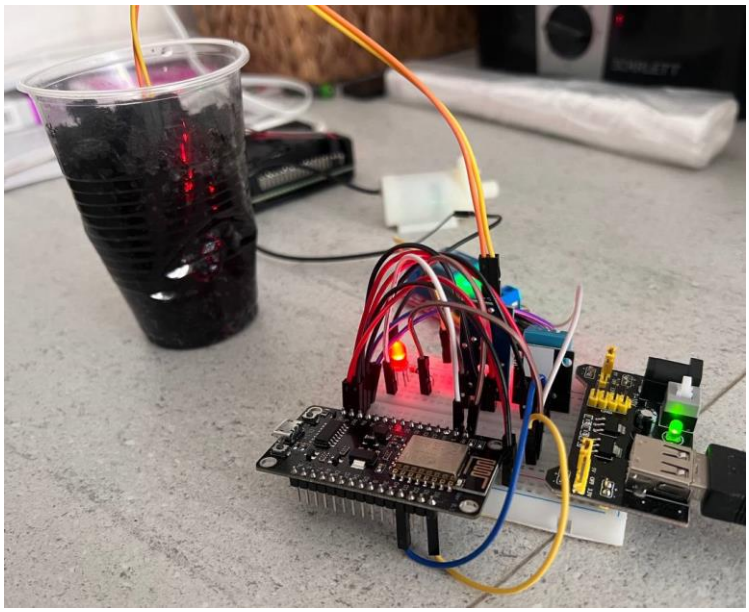
Описание подключения компонентов:

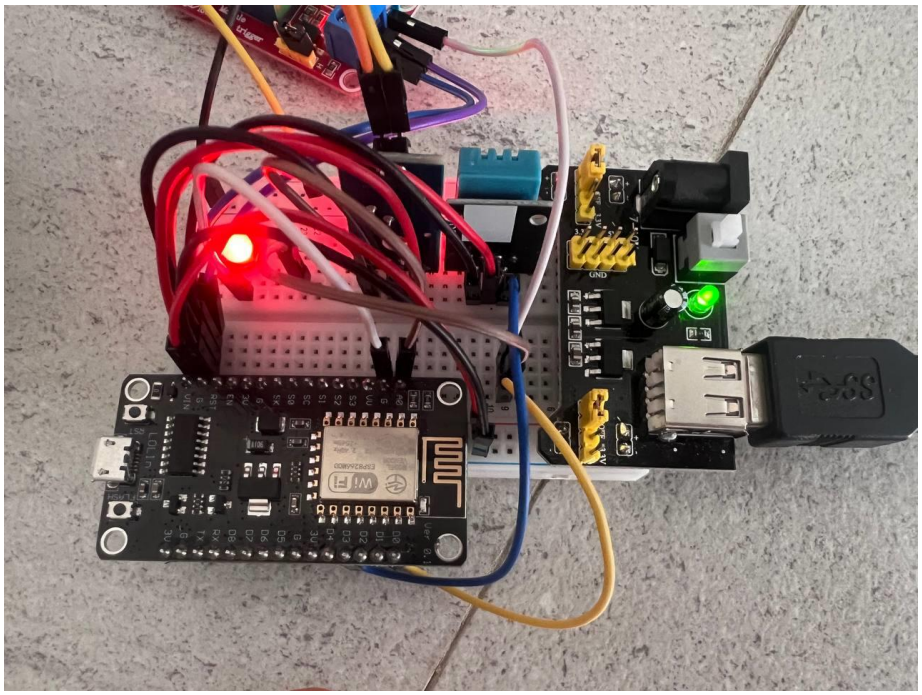
- Источник питания в 5 Вольт подключается к пинам VCC(+) и GND(-) микроконтроллера NodeMCU V3.
- Контакты +, - и S датчика DHT11 подключаются к пинам VCC, GND и D3 соответственно.
- Контакты VCC, GND и A0 датчика YL-38 подключаются к пинам VCC, GND и A0 соответственно.
- Контакты DC+, DC- и IN Реле 5v10A подключаются к пинам VCC, GND и D0 соответственно.

- Параллельно к пину D0 через резистор вешаем светодиод для индикации работы насоса.
- Через отдельный источник питания в 5v подключается насос для воды. + подключается в разрыв между контактами реле NC и COM.

В итоге, после сборки получилось вот такое устройство:



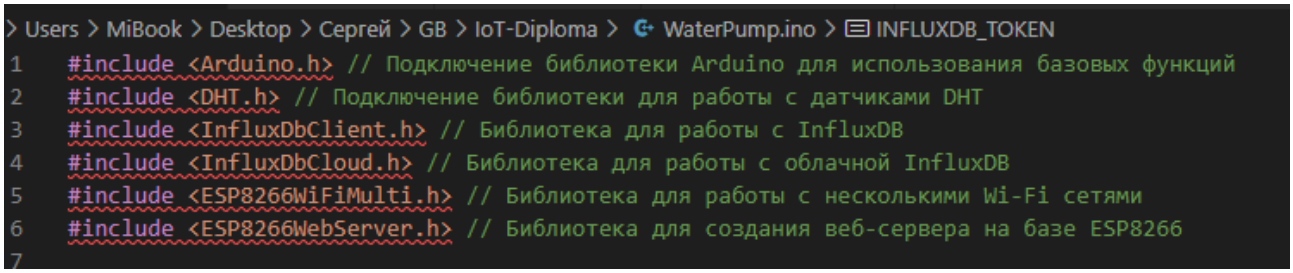




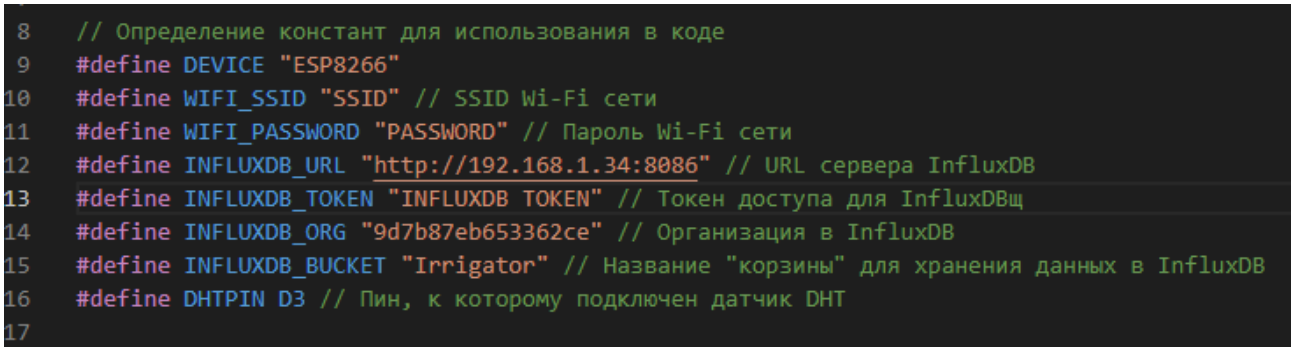
2.3 Программирование системы

Как я уже писал выше, в качестве среды разработки был выбран Arduino IDE. Но в VS Code код выглядит более понятно и читаемо, поэтому демонстрация кода будет в скриншотах из этой программы.

Для начала подключаю необходимые библиотеки:



Далее определяю необходимые константы:



Следующим шагом задаю необходимые переменные, инициализирую объекты и определяю пины, к которым подключены датчики:

```
// Инициализация объекта DHT
DHT dht(DHTPIN, DHT11);

// Инициализация объекта для работы с несколькими Wi-Fi сетями
ESP8266WiFiMulti wifiMulti;

// Инициализация HTTP сервера на порту 80
ESP8266WebServer server(80);

// Создание клиента для работы с InfluxDB
InfluxDBClient clientdb(INFLUXDB_URL, INFLUXDB_ORG, INFLUXDB_BUCKET, INFLUXDB_TOKEN, InfluxDbCloud2CACert);

// Создание объекта Point для отправки данных в InfluxDB
Point sensor("measurements");

// Определение пинов для сенсоров и насосов
const int moisturePin = A0; // Пин для датчика влажности почвы
const int waterPumpPin = D0; // Пин для управления насосом

// Переменные состояния
bool waterPumpOn = false; // Состояние насоса (включен/выключен)
bool automaticMode = true; // Режим автоматического полива по умолчанию

// Интервалы для отправки данных и проверки состояния
const unsigned long interval = 10000; // Интервал отправки данных в InfluxDB (10 секунд)
const unsigned long interval1 = 1000; // Интервал для проверки влажности почвы (1 секунда)

// Переменные для хранения значений датчиков
float m = 0; // Влажность почвы
float h = 0; // Влажность воздуха
float t = 0; // Температура воздуха

// Временные переменные для управления интервалами
unsigned long previousMillis = 0;
unsigned long previousMillis1 = 0;

```

Функция void setup():

```
// Инициализация модулей и создание HTTP сервера
void setup() {
  Serial.begin(115200); // Начало серийной связи на скорости 115200 бод
  pinMode(waterPumpPin, OUTPUT); // Установка пина насоса как выход
  digitalWrite(waterPumpPin, HIGH); // Установка начального состояния насоса (ВЫКЛ)

  dht.begin(); // Инициализация датчика DHT

  WiFi.mode(WIFI_STA); // Установка режима работы Wi-Fi
  wifiMulti.addAP(WIFI_SSID, WIFI_PASSWORD); // Добавление Wi-Fi сети

  // Подключение к Wi-Fi
  Serial.print("Connecting to WiFi");
  while (wifiMulti.run() != WL_CONNECTED) { // Ожидание подключения к Wi-Fi
    Serial.print(".");
    delay(500);
  }
  Serial.println("\nWiFi connected"); // Соединение установлено

  // Проверка подключения к InfluxDB
  if (clientdb.validateConnection()) {
    Serial.print("Connected to InfluxDB: ");
    Serial.println(clientdb.getServerUrl());
  } else {
    Serial.print("InfluxDB connection failed: ");
    Serial.println(clientdb.getLastErrorMessage());
  }

  // Установка обработчиков HTTP запросов
  server.on("/", HTTP_GET, handleRoot); // Обработчик для корневого URL
  server.on("/toggle-pump", HTTP_POST, [](){ // Обработчик для переключения насосов
    waterPumpOn = !waterPumpOn; // Переключение состояния насоса
    digitalWrite(waterPumpPin, waterPumpOn ? HIGH : LOW); // Установка состояния насоса
    server.sendHeader("Location","/"); // Перенаправление на главную страницу
    server.send(303); // Отправка кода ответа
  });

  server.on("/toggle-mode", HTTP_POST, [](){ // Обработчик для переключения режима
    toggleMode(); // Вызов функции для смены режима
    server.sendHeader("Location","/"); // Перенаправление на главную страницу
    server.send(303); // Отправка кода ответа
  });

  // Запуск HTTP сервера
  server.begin();
  Serial.println("HTTP server started"); // Успешный запуск сервера
}
```

Функция для отправки данных в InfluxDB:

```
// Функция для отправки данных в InfluxDB и ожидание
void sendInfluxDB() {
  sensor.clearFields(); // Очистка предыдущих полей данных
  sensor.addField("Temperature C", t); // Добавление температуры
  sensor.addField("Humidity %", h); // Добавление влажности воздуха
  sensor.addField("Moisture %", m); // Добавление влажности почвы

  // Отладочная информация
  Serial.print("Writing: ");
  Serial.println(clientdb.pointToLineProtocol(sensor));
  Serial.print("Moisture Percentage: ");
  Serial.print(m);
  Serial.print("%. Temperature: ");
  Serial.print(t);
  Serial.print(" C, Humidity: ");
  Serial.print(h);
  Serial.println("%. Sent to InfluxDB.");

  // Проверка соединения с Wi-Fi
  if (wifiMulti.run() != WL_CONNECTED) {
    Serial.println("WiFi connection lost");
  }

  // Запись данных в InfluxDB
  if (!clientdb.writePoint(sensor)) {
    Serial.print("InfluxDB write failed: ");
    Serial.println(clientdb.getLastErrorMessage());
  }

  Serial.println("Waiting for 8 seconds..."); // Ожидание перед следующей отправкой
  delay(8000); // Задержка в 8 секунд
}
```

HTTP сервер и переключение между режимами работы:

```
// Обработка HTTP запроса на корневой странице
void handleRoot() {
  // Формирование HTML-страницы
  String page = "<html lang='ru'><head><meta charset='UTF-8'><style>button{padding:10px 20px;font-size:16px;}</style></head><body>";
  page += "<h1>Система автоматического полива (beta)</h1>";
  page += "<p>Температура воздуха: " + String(t) + "C</p>";
  page += "<p>Влажность воздуха: " + String(h) + "%</p>";
  page += "<p>Влажность почвы: " + String(m) + "%</p>";
  page += "<form action='/toggle-pump' method='post'>"; // Форма для управления насосом
  page += "<button type='submit' style='background-color:#4CAF50;color:white;padding:10px 20px;font-size:16px;border:none;cursor:pointer;'>";
  page += waterPumpOn ? "Включить полив" : "Выключить полив"; // Текст кнопки в зависимости от состояния насоса
  page += "</button>";
  page += "</form>";
  page += "<form action='/toggle-mode' method='post'>"; // Форма для переключения режимов
  page += "<button type='submit' style='background-color:#f44336;color:white;padding:10px 20px;font-size:16px;border:none;cursor:pointer;'>";
  page += automaticMode ? "Ручной режим" : "Автоматический режим"; // Текст кнопки в зависимости от режима
  page += "</button>";
  page += "</form>";
  page += "</body></html>";

  server.send(200, "text/html", page); // Отправка HTML-страницы клиенту
}

// Переключение между ручным и автоматическим режимами управления насосом
void toggleMode() {
  automaticMode = !automaticMode; // Переключение режима
  if (!automaticMode) {
    digitalWrite(waterPumpPin, HIGH); // Если в ручном режиме, выключаем насос
  }
  handleRoot(); // Обновление корневой страницы
}
```

Основной цикл выполнения программы:

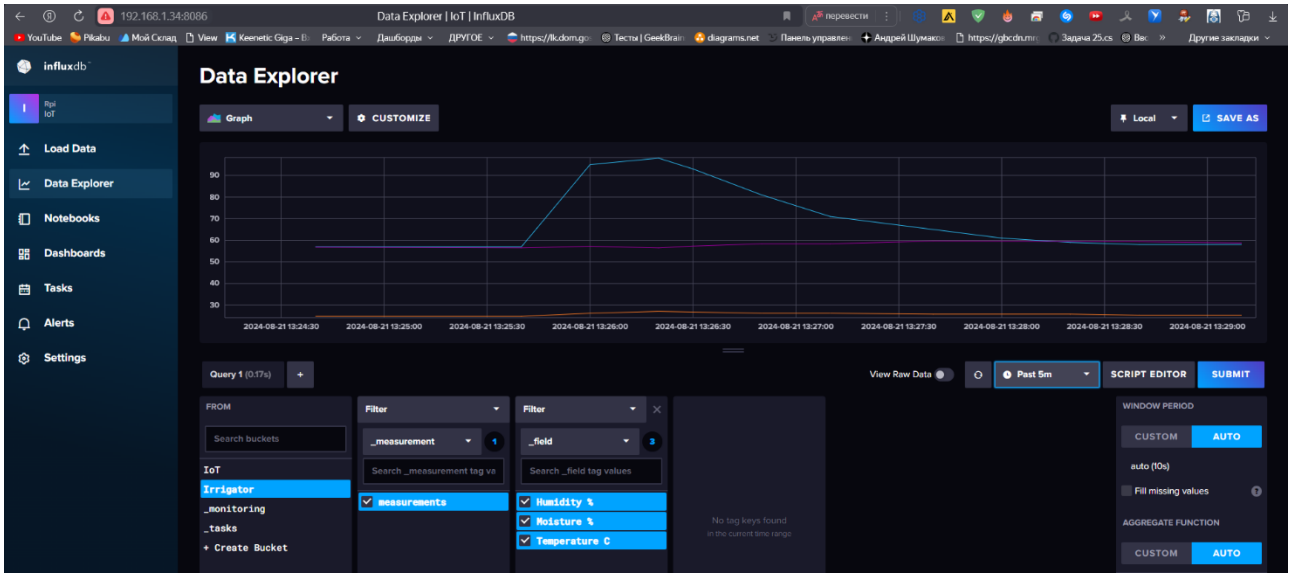
```
// Основной цикл выполнения программы
void loop() {
  unsigned long currentMillis = millis(); // Получение текущего времени

  server.handleClient(); // Обработка входящих HTTP-запросов

  // Чтение данных с датчиков
  h = dht.readHumidity(); // Чтение влажности
  t = dht.readTemperature(); // Чтение температуры
  m = 100.00 - ((analogRead(moisturePin) / 1023.00) * 100.00); // Чтение влажности почвы

  // Вывод данных о влажности почвы в сериал
  if ((unsigned long)(currentMillis - previousMillis1) >= interval1) {
    Serial.print("Soil Moisture: ");
    Serial.print(m);
    Serial.println("%");
    previousMillis1 = millis(); // Обновление времени последней записи
  }

  // Автоматический режим управления насосом
  if (automaticMode) {
    // Логика управления насосом на основе влажности почвы
    if (m < 50) {
      digitalWrite(waterPumpPin, LOW); // Включаем насос, если влажность низкая
    } else if (m >= 50 && m < 55) {
      digitalWrite(waterPumpPin, LOW); // Включаем насос для поддержания влажности
    } else {
      digitalWrite(waterPumpPin, HIGH); // Выключаем насос, если влажность достаточная
    }
  }

  // Отправка данных в InfluxDB по истечении интервала
  if ((currentMillis - previousMillis) >= interval) {
    sendInfluxDB(); // Вызов функции отправки данных
    previousMillis = millis(); // Обновление времени последней отправки
  }
}
```

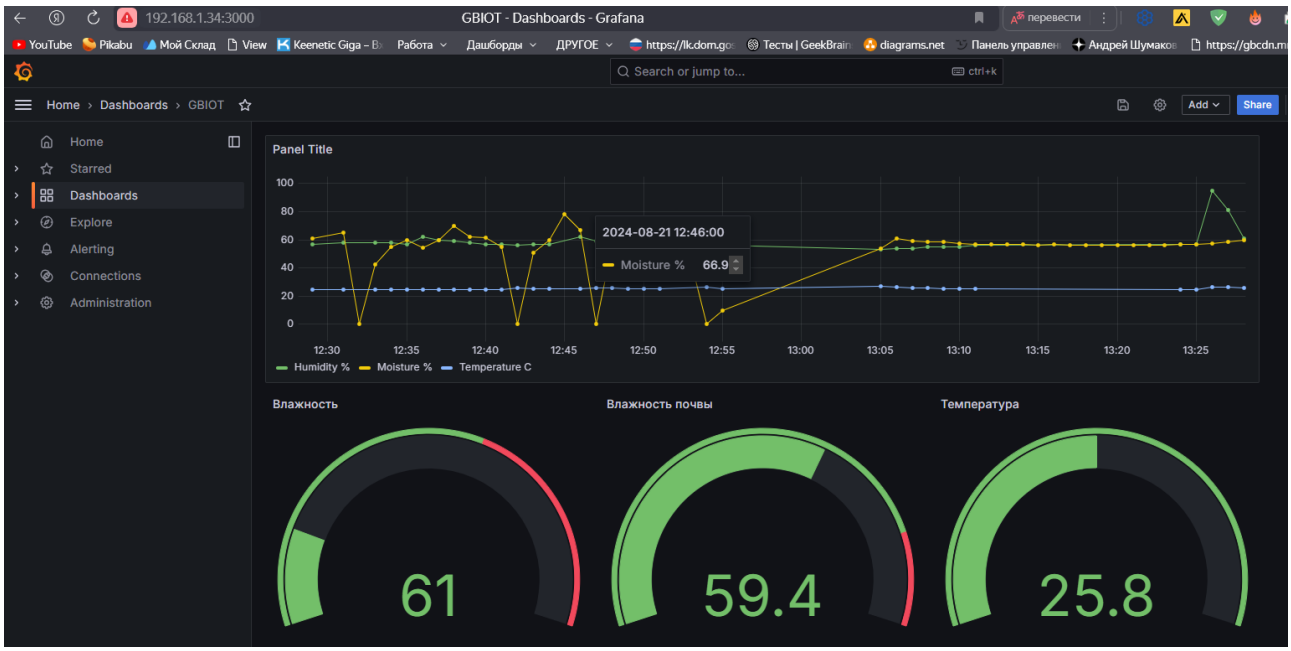
2.4 Тестирование

В результате тестирования устройства было принято решение отказаться от отправки данных сразу в InfluxDB, а использовать протокол MQTT и отправлять данные с датчиков сначала в Node-RED, а оттуда уже дальше в InfluxDB. Поэтому финальный код был переработан и сокращен.

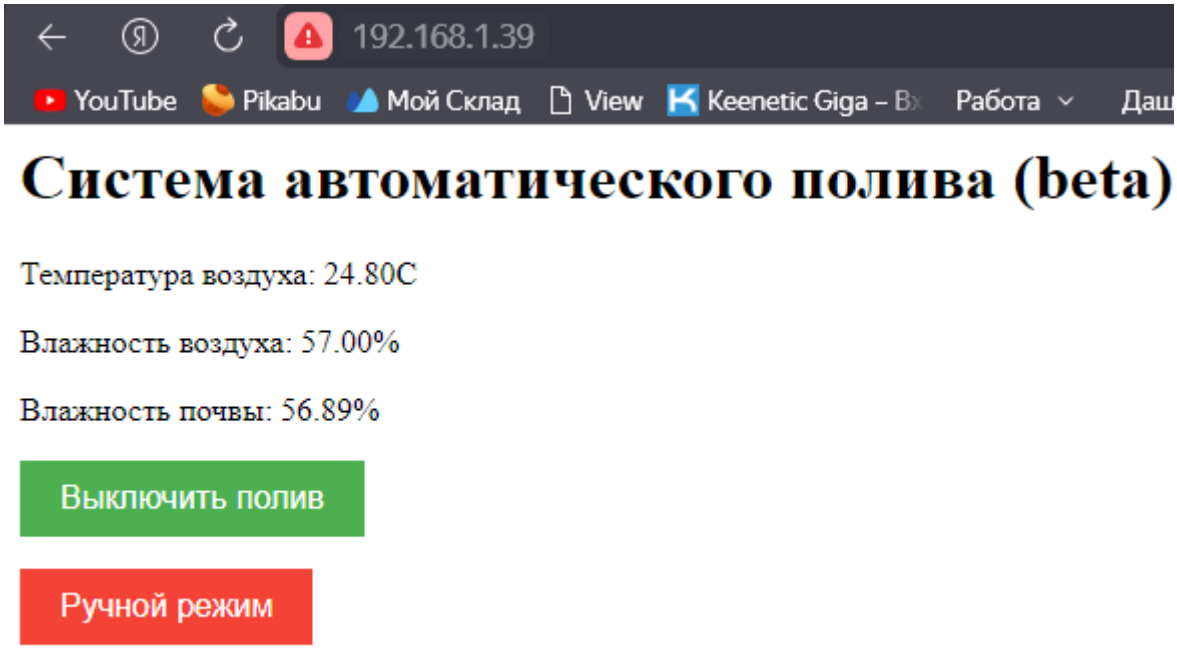
Полный код скетча с подробными комментариями можно посмотреть по ссылке: Irrigator.ino.

Данные с устройства успешно выгружаются и сохраняются в базе данных и передаются в Grafana. Ниже представлены скриншоты работы:

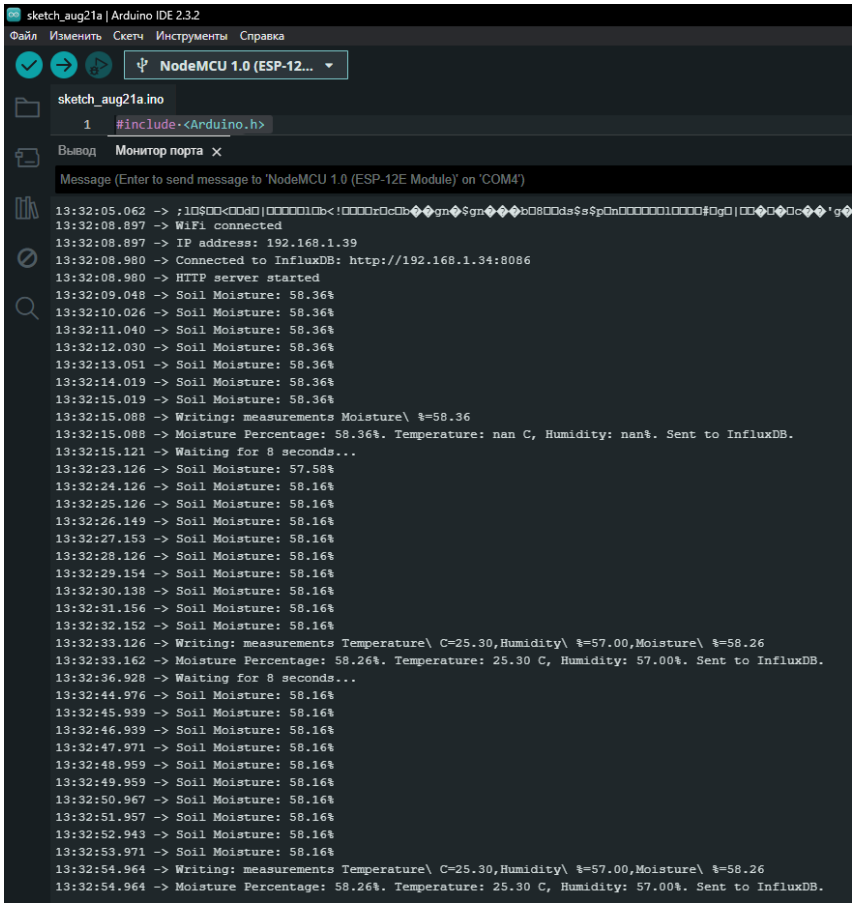




HTTP сервер успешно запускается и система управляется через запросы с сервера:



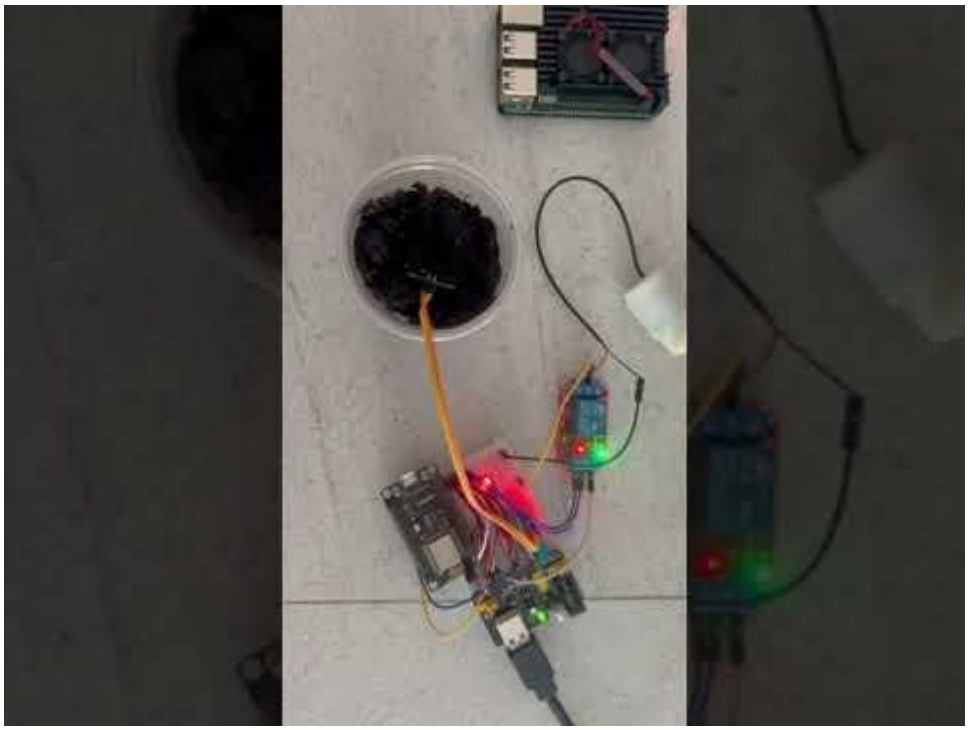
Ниже представлен пример работы монитора порта устройства:



Также выяснилось, что недостаток мощности вызывает отказ в срабатывании реле, решилось установкой более мощного блока питания для водяного насоса.

Для корректной работы в реальных условиях необходимо приобрести более дорогостоящие датчики, более производительный насос, который питается от сети 220 Вольт. В будущем планируется дооснастить систему приемником 443 МГц, для управления с радиопульта.

Демонстрация работы устройства представлена в видео:



Заключение

В процессе разработки системы автоматического полива для дачного участка была создана эффективная и инновационная система, способная обеспечить оптимальные условия для роста растений. Проектная работа относится к сфере Интернет вещей (IoT) и включает в себя широкий спектр технологий, таких как использование микроконтроллера ESP8266, датчиков температуры и влажности, а также информационных протоколов MQTT для передачи данных.

Основные преимущества системы заключаются в:

Автоматизации процесса полива, что значительно экономит время и ресурсы, особенно актуально для пользователей, не имеющих возможности регулярно посещать свои участки.

Удаленном управлении и мониторинге состояния растений в режиме реального времени, что позволяет принимать оперативные решения по уходу за растениями.

Интеллектуальном расходе воды, что способствует более эффективному использованию ресурсов и минимизации их потерь.

Реализованная система является примером современного подхода к агрономии, что открывает новые возможности для оптимизации сельскохозяйственного производства и ухода за растениями в домашних условиях.

Список используемой литературы

1. Алексеев, И. И. "Интернет вещей: Основы и применение". — Москва: Научное издательство, 2018.
2. Дьяков, С. С. "Микроконтроллеры: от А до Я". — Санкт-Петербург: Издательство "БХВ-Петербург", 2020.
3. Петров, В. Н., Михайлов, А. Г. "Системы автоматизации в агрономии". — Москва: Издательство "АГРО", 2019.
4. "Datasheet DHT11 Sensor". https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT11.pdf
5. "ESP8266 WiFi Module Documentation". https://www.esp8266.com/
6. "MQTT Protocol Specification". http://mqtt.org/
7. "Герасименко, И. А. "Технологии и инструменты для разработки IoT приложений". — Минск: РИПОЛ классик, 2021.