**Національний технічний університет України**

**«Київський політехнчний інститут імені Ігоря Сікорського»**

**Навчально-науковий інститут атомної та теплової енергетики**

**Кафедра цифрових технологій в енергетиці**

**Звіт**

**з лабораторної роботи №6**

**з дисципліни «Розробка застосунків Інтернету речей та сенсорних мереж»**

Варіант №16

Виконав:

студент групи ТР-23

Ровний Г.О.

Дата здачі: 20.10.2025

КИЇВ – 2025

**Мета роботи:** Навчитися зчитувати поточний стан сенсорів і приймати значення лічильника, аналізувати й відображати отримані данні у користувацькому застосунку.

**Поставлене завдання:**

1) Проаналізувати всі вимоги до апаратної частини.

2) Розробити схему апаратного забезпечення агента і програмного забезпечення пристрою.

3) Вибрати програмне забезпечення.

4) Розробити програмно-апаратний комплекс.

5) Продемонструвати функції створеного комплексу.

6) Оформити звіт за результатами виконання завдання. Включити у звіт скрини і надати пояснення.

**Результат виконання роботи**

За умовами завдання треба розробити програмно-апаратний комплекс. Подано схему, створену після проведення аналізу вимог до апаратної частини, взята із завдання до лабораторної роботи:

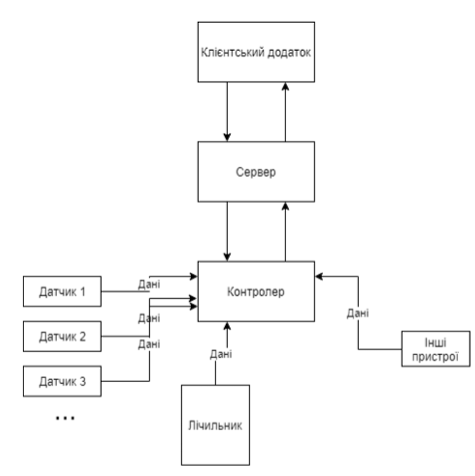


Рис. 6.1. Схема апаратного забезпечення агента

Сервер і контролер — складові апаратної частини. Контролер взаємодіє датчиками та іншими пристроями, керуючи системними елементами. Отримані дані передаються серверові, який виконує аналіз і висуває відповідні команди контролеру. Цей сервер є одноплатним комп’ютером, який не лише налаштовує передачу даних за допомогою TCP/IP-протоколу, але й має можливість підключення пристроїв через UART. Це надає можливість налаштувати обмін даними між сервером і клієнтським додатком. Клієнтський додаток, який працює на операційній системі Windows, є лише інтерфейсом для візуалізації отриманих даних. Сервер аналізує та обробляє дані, які надходять від контролера, і висилає відповіді на них. Така взаємодія через TCP/IP і UART дає можливість системі ефективно обмінюватися інформацією і приймати команди для керування. Загальна структура системи передбачає збір даних на контролері, їхню передачу до сервера, аналіз і відправку команд, а також відображення результатів на клієнтському додатку. Цей комплексний підхід дає можливість забезпечити ефективне функціонування і відстеження стану системи

****

Рис. 6.2. Схема програмного забезпечення пристрою

Контролер має за завдання зчитувати показники приєднаних пристроїв щокожні секунди і реагувати на команди, які надсилає сервер. Сервер виступає як головна частина агента і має надійно функціонувати, навіть у випадку втрати зв’язку з іншим клієнтом, забезпечуючи безперервну роботу програми.

У проєктуванні сервера реалізовано його як агент, який одночасно є сервером і клієнтом. Сервер відкриває порт для прослуховування інших 127 клієнтів та очікує на з’єднання для отримання даних. Коли агент має відправити дані, він створює тимчасового TCP-клієнта, який здійснює підключення, передає дані та видаляється. Цей метод комунікації забезпечує стабільність і надійність, навіть якщо агент втратив зв’язок з іншим агентом.

Такий підхід не тільки гарантує стійку роботу, але й дає можливість обмінюватися даними одночасно з різними агентами. Клієнт-додаток використовується виключно для візуалізації отриманих даних від агента і як засіб комунікації з ним.

**Вибір апаратного забезпечення.**

Для налаштування контролера використовується середовище розробки STM32CubeMX, яке є зручним графічним інструментом для конфігурації мікроконтролерів STM32 та мікропроцесорів. Це середовище дозволяє швидко та інтуїтивно задавати параметри мікроконтролера, використовуючи покроковий підхід до налаштування периферійних модулів, таймерів, інтерфейсів введення-виведення та енергоспоживання. Генерація коду здійснюється автоматично для ядра Arm® Cortex®-M, що значно спрощує розробку програмного забезпечення та мінімізує помилки на етапі ініціалізації. Для більш складних пристроїв можливе часткове створення дерева пристроїв для Linux® на основі ядра Arm® Cortex®, що забезпечує більшу гнучкість та масштабованість при розгортанні на потужніших процесорах.

Для написання коду контролера використовується середовище розробки System Workbench for STM32, яке забезпечує повноцінне середовище для програмування, компіляції та налагодження додатків для мікроконтролерів STM32. Це середовище підтримує інтеграцію з STM32CubeMX, дозволяючи імпортувати згенеровані налаштування без додаткових маніпуляцій. У процесі розробки активно використовуються бібліотеки HAL (Hardware Abstraction Layer), які надають високорівневі інтерфейси для роботи з апаратною частиною контролера, спрощуючи управління периферійними пристроями. Для роботи з одноплатним комп’ютером застосовується бібліотека WiringOp, що дозволяє ефективно керувати його апаратними можливостями, такими як GPIO, I²C, SPI та інші інтерфейси.

Програмне забезпечення контролера реалізується на мові програмування C, що забезпечує високу продуктивність, ефективне використання ресурсів та можливість низькорівневого управління мікроконтролером. Для одноплатного комп’ютера використовується мова C++, яка дозволяє створювати більш структурований та об’єктно-орієнтований код, що спрощує масштабування проєкту та його підтримку. Клієнтський додаток розробляється на мові C#, що дає змогу створювати зручний графічний інтерфейс для взаємодії з користувачем, включаючи відображення отриманих даних та можливість керування пристроями. Використання C# також забезпечує інтеграцію з сучасними серверними технологіями та базами даних, що дозволяє легко обробляти великі обсяги інформації.

Таким чином, вибір засобів розробки є оптимальним для реалізації системи, забезпечуючи стабільну роботу всіх компонентів, ефективну обробку даних та зручний інтерфейс для взаємодії користувача з пристроями.

**Демонстрація функції створеного комплексу.**

В інтерфейсі користувача програмного комплексу передбачено три основні вкладки: «Stats», «Graph» та «Console». Кожна з них виконує окрему функцію та дозволяє взаємодіяти з даними системи в режимі реального часу.

Вкладка «Stats» призначена для візуалізації поточного стану електромережі. Вона містить схему системи, на якій у відповідних точках розташовані датчики струму. Для зручності сприйняття дані датчиків відображаються у вигляді кольорових чотирикутників. Якщо датчик не фіксує струм у певній точці, його рамка забарвлюється в червоний колір, що сигналізує про відсутність електропостачання. Якщо ж датчик реєструє струм, його індикатор стає яскраво-зеленим. Окрім цього, на вкладці відображається стан вентиляційної установки. Поруч із відповідним позначенням міститься чотирикутний індикатор, який показує, чи увімкнена система. Також передбачено текстове поле, в якому відображається поточний рівень подачі струму на установку у відсотковому значенні:

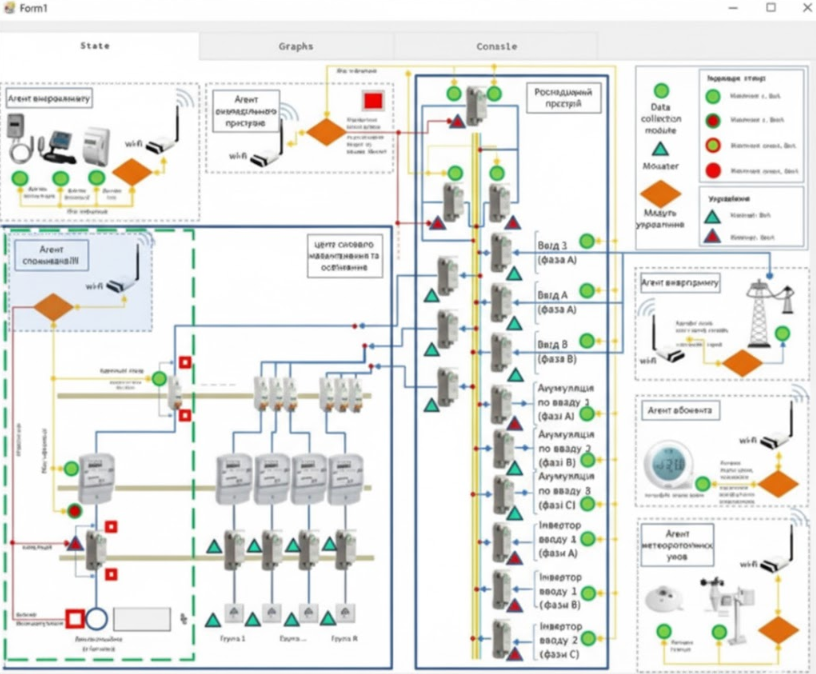


Рис 6.4. Stats

Вкладка «Graph» містить графік навантаження електромережі. Вона забезпечує динамічний перегляд змін у роботі системи, адже дані оновлюються в реальному часі. Графік відображає значення лічильника, що змінюються кожну секунду, дозволяючи контролювати споживання електроенергії та оцінювати стабільність роботи системи. Такий підхід дає можливість оперативно аналізувати навантаження та приймати відповідні рішення щодо його коригування.

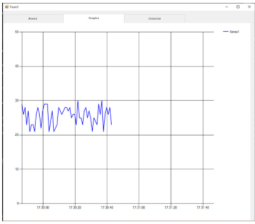


Рис. 6.5. Вкладка графіка

Вкладка «Console» виконує роль інструменту для тестування роботи агента. Вона складається з трьох основних блоків: ручного керування, автоматичного керування та консолі подій. Консоль фіксує всі зміни у системі, записує можливі помилки та дозволяє відстежувати ключові події. Це дає змогу швидко знаходити несправності та аналізувати причини їх виникнення. У блоці ручного керування користувач може ввімкнути або вимкнути вентиляційну установку, а також змінити рівень її потужності. Всі команди передаються агенту через TCP/IP-з’єднання після натискання відповідної кнопки. У разі невдалої спроби з’єднання консоль реєструє помилку та надає її опис. Автоматичне керування дозволяє задавати часові інтервали для роботи вентиляційної установки. Користувач може додавати або видаляти часові проміжки, а всі внесені зміни миттєво відображаються у списку та консолі. У разі некоректного введення часу система видає повідомлення про помилку та очікує правильного вводу.

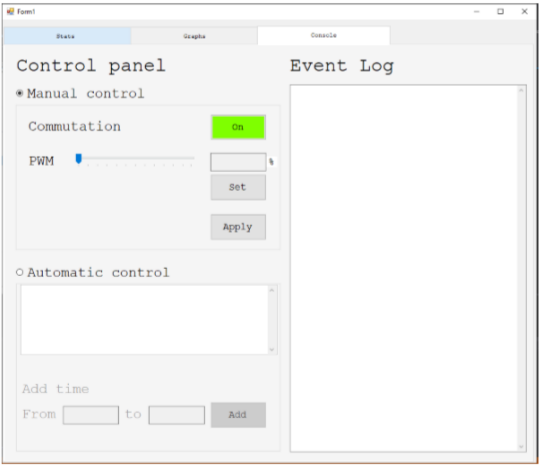


Рис. 6.6. Консольна вкладка

Ручний контроль забезпечує можливість безпосереднього керування вентиляційною установкою в режимі реального часу. Користувач може вручну вмикати або вимикати систему, а також встановлювати рівень її потужності, задаючи відповідні параметри. Після внесення змін команда надсилається агенту через TCP/IP-з’єднання, що дозволяє миттєво застосувати обрані налаштування. Такий підхід надає повний контроль над роботою установки та дозволяє гнучко реагувати на поточні умови експлуатації.

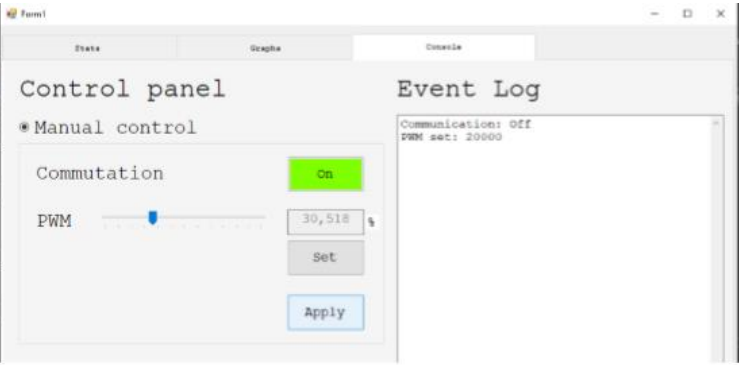


Рис. 6.7. Ручний контроль

Опрацювання помилок є важливою частиною функціоналу системи, що дозволяє контролювати коректність її роботи та усувати можливі збої. У випадку виникнення помилки система автоматично реєструє її в журналі подій та надає опис причини. Це дає змогу швидко визначити, чому сталася несправність, наприклад, якщо клієнт не зміг підключитися до агента через його відключення від мережі. Такий підхід підвищує стабільність роботи та спрощує діагностику можливих проблем.

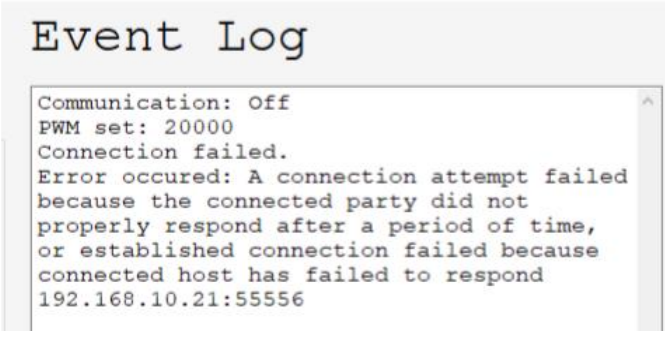


Рис. 6.8. Опрацювання помилок

Автоматичний контроль дозволяє користувачеві задати часові інтервали, в які вентиляційна установка має вмикатися або вимикатися без необхідності ручного втручання. Для цього передбачена спеціальна форма, яка дозволяє ввести бажаний час і перевіряє його правильність перед додаванням у список інтервалів. Такий механізм забезпечує автоматизацію роботи та дозволяє налаштувати систему для роботи відповідно до встановленого графіка, що зменшує потребу в постійному контролі з боку користувача.

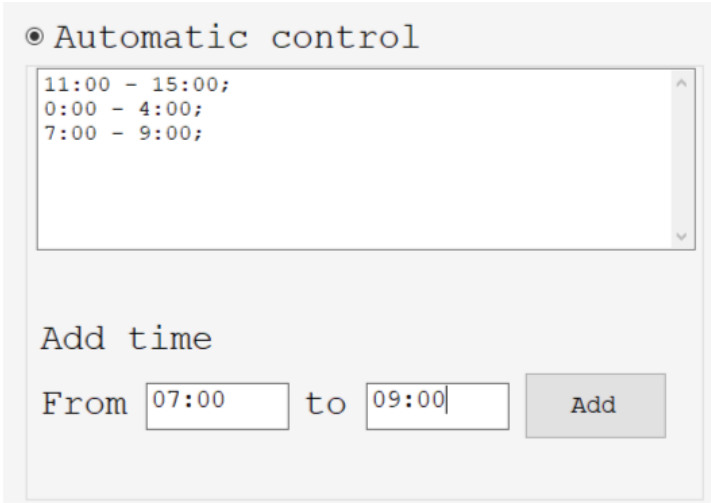


Рис. 6.9. Автоконтроль

Відображення змін у консолі дозволяє фіксувати всі дії, що відбуваються у системі, забезпечуючи прозорість її роботи. Після внесення змін до списку часових інтервалів або ручного регулювання параметрів установки відповідні записи з’являються в журналі подій. Це дозволяє користувачеві переглядати історію змін та переконатися в правильності виконання команд. Така функція є корисною для моніторингу роботи та аналізу ефективності внесених налаштувань.

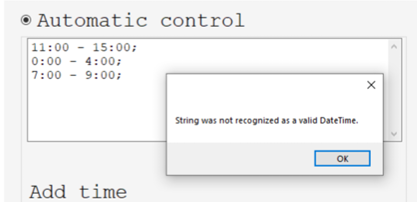


Рис. 6.10. Відображення змін у консолі

Запис усіх подій до файлу дозволяє вести детальний журнал роботи системи, що зберігається у вигляді текстового документа із зазначенням дати та часу виконання операцій. Кожен етап функціонування програми фіксується у файлі, що дозволяє відстежувати її поведінку та аналізувати можливі збої. У випадку несподіваного завершення роботи це дає можливість встановити причину зупинки та оперативно відновити систему без втрати важливих даних:

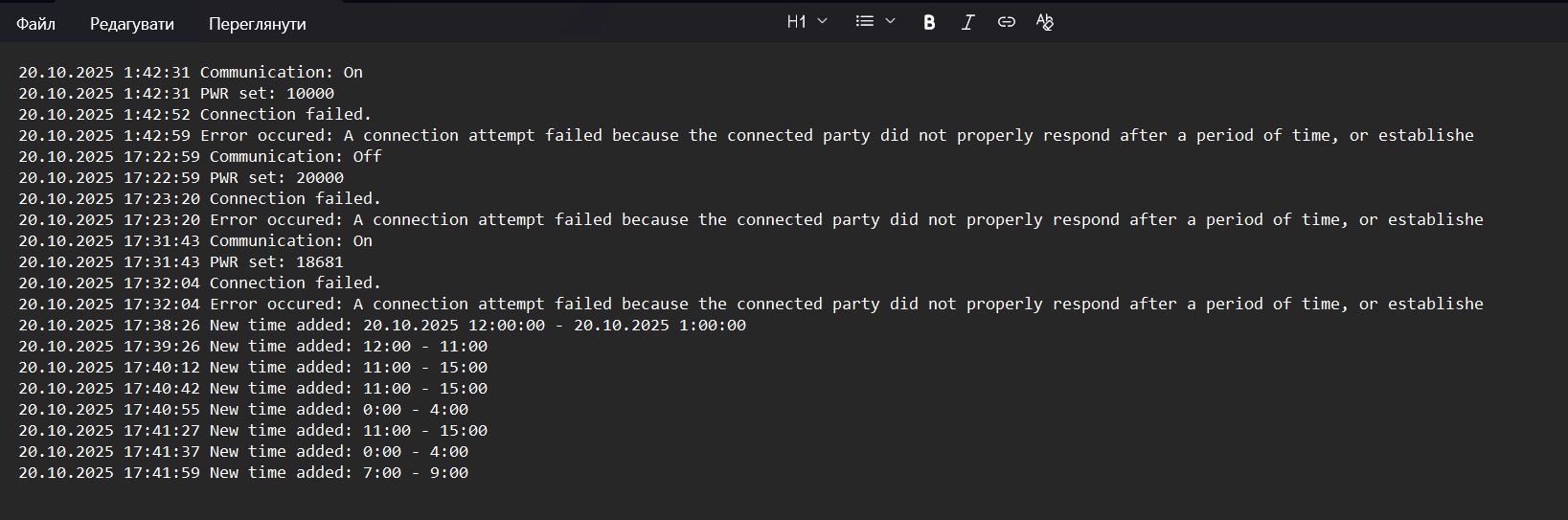


Рис. 6.11. Запис усіх подій до файлу

**Висновок:**

У результаті виконання реалізовано систему моніторингу аналогових сенсорів і лічильника за допомогою мікроконтролера STM32F103C8T6. Програмне забезпечення дає можливість отримувати дані та відправляти команди обладнанню. Сконструйовано апаратно-програмний агент на базі Orange Pi та контролера сімейства STM32. Програмна система має клієнт-серверну архітектуру.