

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему: ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ
ВІДДАЛЕНОГО МОНІТОРИНГУ СПОЖИВАННЯ ЕЛЕКТРОЕНЕРГІЇ
RESEARCH AND SOFTWARE IMPLEMENTATION OF A SYSTEM FOR
REMOTE MONITORING OF ELECTRICITY CONSUMPTION

Виконав: студент 2 курсу, групи КНТ-212м

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Системи штучного інтелекту

ТРОЦ Є. С.

(ПРИЗВИЩЕ та ініціали)

Керівник ПАРХОМЕНКО А.В.

(ПРИЗВИЩЕ та ініціали)

Рецензент ЗЕЛЕНЬОВА І.Я.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Системи штучного інтелекту
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2023 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ТРОЦА Євгенія Сергійовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація системи віддаленого моніторингу споживання електроенергії. Research and Software Implementation of a System for Remote Monitoring of Electricity Consumption
керівник проєкту (роботи) к.т.н., доцент, ПАРХОМЕНКО Анжеліка Володимирівна

(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “14” листопада 2023 року № 437

2. Строк подання студентом проєкту (роботи) 1 грудня 2023 року

3. Вихідні дані до проєкту (роботи) рекомендована література, завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Дослідження та вибір технологій та засобів прототипування. 3. Розробка та дослідження прототипу системи віддаленого моніторингу споживання електроенергії. 4. Керівництво програміста. 5. Керівництво користувача.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	ПАРХОМЕНКО А.В., доцент		
Нормоконтроль	БЄЛОВА А.В., асистент		

7. Дата видачі завдання “ 8 ” вересня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2 тиждень	Розділ 1
3	Дослідження та вибір програмного та апаратного забезпечення.	3 тиждень	Розділ 2
4	Розробка програмно-апаратного прототипу, тестування та експериментальне дослідження.	4-6 тижні	Розділ 3
5	Написання керівництва програміста	7 тиждень	Розділ 4
6	Написання керівництва користувача	8 тиждень	Розділ 5
7	Оформлення пояснювальної записки та документів до неї.	9-10 тижні	Додатки
8	Нормоконтроль та рецензування.	10-11 тижні	
9	Захист роботи.	12 тиждень	

Студент(ка)

(підпис) Євгеній ТРОЦ
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Анжеліка ПАРХОМЕНКО
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
124 с., 16 табл., 87 рис., 2 дод., 42 джерела.

ЕЛЕКТРОСПОЖИВАННЯ, ОПТИМІЗАЦІЯ, МОНІТОРИНГ, MQTT, ARDUINO, ESP8266, ESP32, ЕНЕРГОЕФЕКТИВНІСТЬ, АЛГОРИТМ, ПРОГРАМА, ПРОТОТИП, РОЗРОБКА.

Об'єкт дослідження – процес віддаленого моніторингу споживання електроенергії.

Предмет дослідження – методи та засоби віддаленого моніторингу споживання електроенергії.

Мета роботи – дослідження та розробка системи віддаленого моніторингу споживання електроенергії для контролю процесу енергоспоживання, підвищення ефективності споживання електроенергії та безпеки при використанні електроприладів.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мова програмування C++, Arduino IDE, платформа програмування Arduino Uno, NodeMCU, комплект розробки M5Stack ATOM на базі WI-FI модуля ESP32, протокол MQTT, платформа OpenHAB, модуль датчика напруги Zmpt101b 250V, трансформатор струму ZMCT103C, модуль електромеханічного реле.

Результати. Розроблено програмне забезпечення, що дозволяє віддалено моніторити процес енергоспоживання та керувати станом роботи електроприладів.

Висновки. Розроблене програмне забезпечення можливо інтегрувати в системи домашньої та офісної автоматизації, що дозволить підвищити ефективність споживання електроенергії та контролювати стан енергосистеми.

Область застосування – системи домашньої та офісної автоматизації.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 124 pages, 87 figures, 16 tables, 2 appendixes, 42 sources.

POWER CONSUMPTION, OPTIMIZATION, MONITORING, MQTT, ARDUINO, ESP8266, ESP32, ENERGY EFFICIENCY, ALGORITHM, PROGRAM, PROTOTYPE, DEVELOPMENT.

Object of study is the process of remote monitoring of electricity consumption.

Subject of study is methods and means of remote monitoring of electricity consumption.

The purpose of the work is research and development of a system for remote monitoring of electricity consumption to control the energy consumption process, increase the efficiency of electricity consumption and safety when using electrical appliances.

Materials, methods, and tools: object-oriented programming, C++ programming language, Arduino IDE, Arduino Uno programming platform, NodeMCU, M5Stack ATOM development kit based on ESP32 WI-FI module, MQTT protocol, OpenHAB platform, sensor module voltage Zmpt101b 250V, current transformer ZMCT103C, electromechanical relay module.

Results. Software has been developed that allows you to remotely monitor the energy consumption process and manage the state of operation of electrical appliances.

Conclusions. The developed software can be integrated into home and office automation systems, which will increase the efficiency of electricity consumption and control the state of the power system.

The field of use is a home and office automation systems.

ЗМІСТ

	С.
Перелік скорочень та умовних познач	9
Вступ	10
1 Аналіз предметної області	11
1.1 Дослідження особливостей системи «Розумний будинок»	11
1.1.1 Аналіз функціоналу системи домашньої автоматизації	11
1.1.2 Огляд структури системи контролю енергозбереження	112
1.2 Огляд особливостей систем Smart metering	15
1.2.1 Основи систем Smart metering	115
1.2.2 Переваги систем Smart metering	116
1.2.3 Принципи роботи систем Smart metering	116
1.3 Огляд функціональних можливостей існуючих засобів заощадження енергії	18
1.3.1 Розумна розетка з лічильником енергоспоживання Ajax Socket	118
1.3.2 Розумна Wi-Fi розетка Sonoff S26	19
1.3.3 Проєкт віддаленого моніторингу енергоефективності та білінгу	20
1.3.4 Розумний лічильник електроенергії з використанням ESP12 та Arduino	22
1.3.5 Інтелектуальний моніторинг споживання електроенергії Raspberry Pi на базі IoT	25
1.3.6 Система моніторингу енергоефективності PEEMS	29
1.3.7 Переваги і недоліки наявних систем заощадження енергії	30
1.4 Постановка задач дипломної роботи	32
2 Дослідження та вибір технологій та засобів прототипування	34
2.1 Програмні засоби підтримки процесу прототипування	34
2.2 Апаратні засоби підтримки процесу прототипування	46
2.2.1 Передмова	46
2.2.2 Параметри приладів для тестування	47

2.2.3 Параметри мікроконтролерів	48
2.2.4 Параметри датчиків та модулів	51
2.3 Висновки до розділу 2.....	57
3 Розробка та дослідження прототипу системи віддаленого моніторингу споживання електроенергії.....	58
3.1 Аналіз вимог до системи.....	58
3.2 Архітектура системи	59
3.3 Розробка та дослідження апаратного забезпечення системи.....	61
3.3.1 Загальна апаратна схема.....	61
3.3.2 Опис реалізації датчика контролю наявності мережевої напруги ..	62
3.3.3 Опис реалізації модуля реле з оптопарою.....	63
3.3.4 Опис реалізації модуля датчика напруги змінного струму Zmpt101b 250V	65
3.3.5 Опис реалізації модуля датчика сили струму ZMCT103C	71
3.3.6 Збирання складових проєкту у прототип	78
3.4 Розробка ПЗ для системи віддаленого моніторингу	83
3.4.1 Алгоритм функціонування.....	83
3.4.2 Функції та змінні програми.....	85
3.5 Висновки до розділу 3.....	90
4 Керівництво програміста	92
4.1 Призначення та умови застосування програмно-апаратного забезпечення	92
4.2 Характеристики програми	92
4.3 Звернення до програми	93
4.4 Вхідні та вихідні дані	94
4.5 Повідомлення	96
5 Керівництво користувача.....	97
5.1 Призначення програмно-апаратного забезпечення.....	97
5.2 Умови виконання програм.....	97
5.3 Виконання програм	98

5.4 Повідомлення	98
Висновки.....	100
Перелік джерел посилань.....	101
Додаток А Текст програми	105
Додаток Б Слайди презентації.....	118

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
BLE	– Bluetooth Low Energy;
IDE	– Integrated Development Environment;
IoT	– Internet of Things;
LED	– Light-Emitting Diode;
MQTT	– Message Queue Telemetry Transport;
SSH	– Secure Shell;
АСКОЕ	– автоматизована система обліку електроенергії;
АЦП	– аналого-цифровий перетворювач;
ДБЖ	– джерело безперебійного живлення;
МК	– мікроконтролер;
ПВП	– первинні вимірювальні перетворювачі;
ПЗ	– програмне забезпечення;
ПК	– персональний комп'ютер;
ШІМ	– широтно-імпульсна модуляція.

ВСТУП

Останнім часом спостерігається тенденція підвищення енергоефективності в різних сферах діяльності людини. Окрему нішу займає домашня автоматизація або «розумний будинок». Така система дозволяє значно підвищити комфорт та зручність від проживання в такому будинку, надає переваги новітніх технологій, а головне – дозволяє заощаджувати електроенергію. Це дозволяє користувачам таких систем значно скоротити витрати на утримання домогосподарства, економити власні кошти, а в перспективі може бути значним кроком на шляху вирішення глобальних проблем заощадження електроенергії.

На сьогоднішній день вже відомо багато технологічних підходів до підвищення енергоефективності. До них відносяться як і прості розетки з таймером, так і складні вироби, що інтегруються до систем домашньої автоматизації, та за допомогою датчиків збирають та аналізують інформацію. Останні окрім функції підвищення енергоефективності також гарантують безпеку в будинку, адже можуть реагувати на коротке замикання, або повідомляти користувачів про наближення до максимально допустимих параметрів використання електроенергії, що може призвести до спрацювання запобіжника, або ж до виникнення пожежі.

Аналіз показав, що програмне та апаратне забезпечення наявних розглянутих пристроїв є недосконалим, а також вимагає модифікації певних компонентів. Тому, тема роботи є актуальною.

Мета роботи – дослідження та розробка системи віддаленого моніторингу споживання електроенергії для контролю процесу енергоспоживання, підвищення ефективності споживання електроенергії та безпеки при використанні електроприладів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Дослідження особливостей системи «Розумний будинок»

1.1.1 Аналіз функціоналу системи домашньої автоматизації

Згідно з проведеними дослідженнями, «Розумний будинок» – це система домашніх пристроїв, здатних виконувати дії та вирішувати певні повсякденні завдання без участі людини. Домашня автоматизація в сучасних умовах – надзвичайно гнучка система, яку користувач конструює та налаштовує самостійно залежно від потреб. Це передбачає, що кожен власник розумного будинку самостійно визначає, які пристрої куди встановити та які завдання вони виконуватимуть. Найбільш поширені приклади автоматичних дій в таких системах – автоматичне ввімкнення та вимкнення світла, автоматична корекція роботи опалювальної системи або кондиціонера та автоматичне повідомлення про вторгнення, загоряння або протікання води [1].

«Розумний будинок» керується за певним алгоритмом. Також забезпечуються всі умови, що дозволяють комплексу надійно виконувати свої функції та безпеки доступу до неї. У приміщенні встановлюються необхідні датчики, що, наприклад, фіксують температуру, якість повітря, відстеження руху, диму, газу та інші елементи керування. При тривалій відсутності рухів для економії енергії може бути відключено світло. Якщо у повітрі високий вміст вуглекислого газу, включається вентиляція [2].

Керуючі елементи приймають сигнали з датчиків та контролюють роботу виконавчих пристроїв, діючи згідно з заданими алгоритмами та поєднуючи наступні системи:

- опалення будинку (за допомогою радіаторів або теплої підлоги);
- вентиляція та кондиціонування;
- охоронна та пожежна сигналізація;
- система контролю доступу;
- контроль аварійних ситуацій: витоку води, газу, аварії в електромережі;

- відеоспостереження (локальне та віддалене);
- управління внутрішнім та вуличним освітленням;
- розподіл відео та аудіо потоків та по приміщеннях (мультирум);
- управління обігрівом зливової каналізації, сходів та доріжок;
- контроль над енергоспоживанням, обмеження пікових навантажень та розподіл навантажень по фазах мережі живлення;
- управління джерелами резервного електроживлення: акумуляторними джерелами безперебійного живлення (ДБЖ) та дизель-генераторами;
- управління каналізаційних насосних станцій та систем автополиву зелених територій;
- управління воротами та шлагбаумами;
- управління шторами, ролетами та жалюзі;
- віддалений моніторинг та керування всіма системами будинку через інтернет [3].

1.1.2 Огляд структури системи контролю енергозбереження

Заощадження енергії — один із факторів зниження витрат на утримання приватного житла, а тому енергозбереження в «розумному будинку» будується на вимкненні світла разом з вашим доглядом, переведенням опалення в економічний режим, відключення розеток, які не потрібні в даний момент. Для додаткової економії реалізовані функції "відслідковувати світло", "відслідковувати звук", відключення систем підігріву підлог у коридорі та санвузлах і т.д [4].

Автоматичні системи керування електропостачанням дозволяють розумно витрачати ресурси, а також забезпечувати повсякденний комфорт та безпеку для всіх членів вашої родини. Наприклад, розумний будинок може сам вимкнути світло за датчиками руху, якщо в кімнаті нікого немає, зменшити

інтенсивність опалення для економії електроенергії, коли на вулиці потепліло, знеструмити групи розеток, вирішивши проблему невимкненої праски [4].

Система управління електрикою в розумному будинку вмикає світло самостійно за датчиком руху або відкриттям дверей, що значно підвищує комфорт користування [4].

Автоматичне керування електропостачанням дозволяє дистанційно вмикати та вимикати електроприлади або налаштувати зручні сценарії відключення розеток [4].

Стабілізатор напруги в системі електропостачання в розумному будинку надійно захищає електроприлади від пошкоджень через стрибки напруги в мережі живлення, а також обладнання забезпечить стабільну роботу найважливіших систем у разі непередбаченого відключення електрики. Крім того, буде негайно надіслано сповіщення про відключення електроенергії. У разі потреби автоматично може бути запущено генератор для заряджання акумуляторів [4].

Таким чином, вирішення проблеми обліку електроенергії вимагає створення автоматизованих систем контролю і обліку, які в загальному випадку містять два або три рівні:

- нижній рівень – первинні вимірювальні перетворювачі (ПВП) з телеметричними виходами, з безперервним або мінімальним інтервалом усереднювання вимірювальних параметрів електроенергії (датчики);

- середній рівень – контролери з вбудованим програмним забезпеченням обліку, які здійснюють в заданому циклі інтервалу усереднювання цілодобовий збір вимірювальних даних з контролерів, накопичення, оброблення і передавання цих даних на верхній рівень, наприклад мікроконтролери (МК) Arduino та ESP;

- верхній рівень – спеціалізоване ПЗ автоматизованої системи обліку електроенергії (АСКОЕ), що здійснює збір інформації з контролера або групи контролерів середнього рівня, підсумково оброблює цю інформацію як по точках обліку, так і по їх групах, відображення і документування даних

обліку у вигляді, зручному для аналізу і керування користувачем (системи домашньої автоматизації, наприклад openHAB) [5].

Рівні сучасних систем моніторингу електропостачання наведено на рис. 1.1 [5].

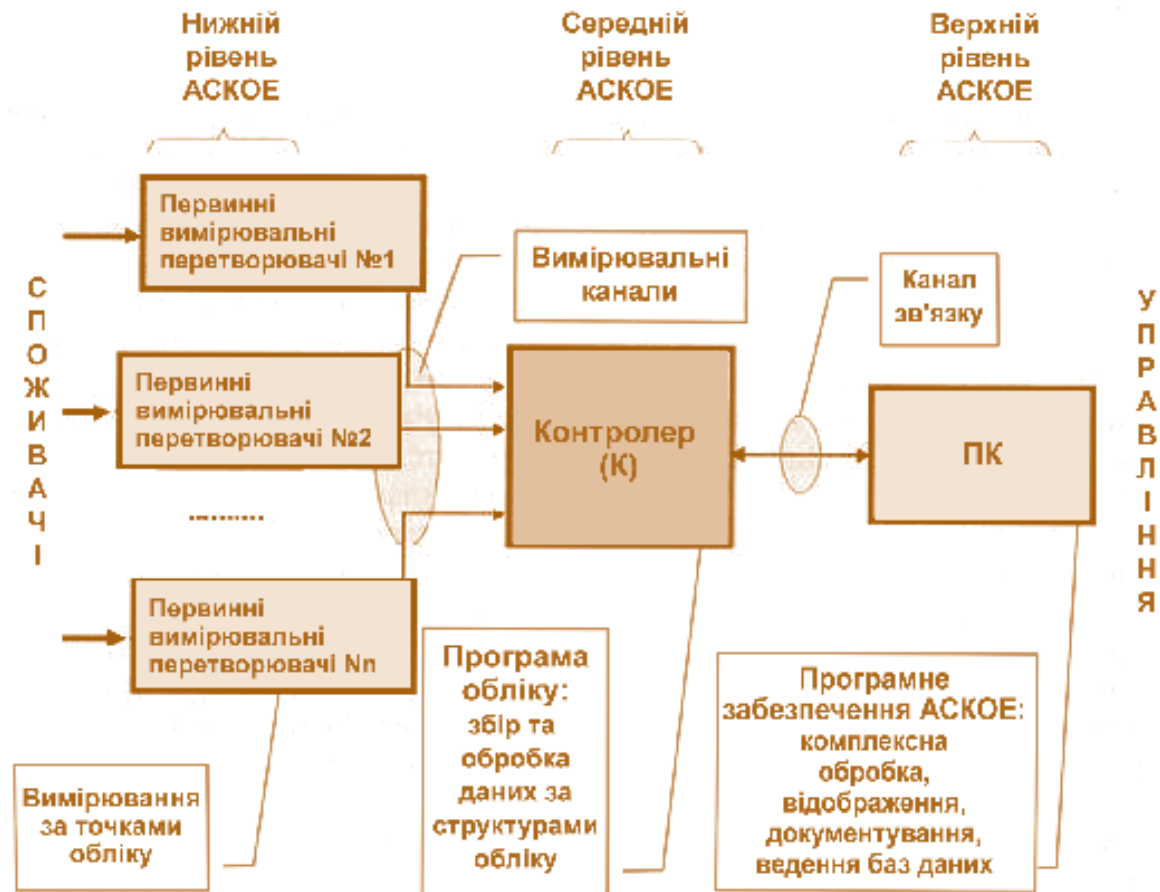


Рисунок 1.1 – Узагальнена схема трьохрівневої автоматизованої системи контролю та обліку енергоресурсів [5]

Було проведено дослідження особливостей існуючих систем домашньої автоматизації, яке показало, що на сьогоднішній день такі системи контролю та обліку енергоресурсів складаються з двох або трьох рівнів, які приймають сигнали з датчиків та контролюють роботу виконавчих пристроїв, що дозволяє моніторити стан енергосистеми.

1.2 Огляд особливостей систем Smart metering

1.2.1 Основи систем Smart metering

Smart metering – це система розумних датчиків, один із сервісів Smart city. Використовується для віддаленого обліку та контролю показників лічильників електро-, газо- та водопостачання, моніторингу теплоенергії та для вчасного виявлення аварійних ситуацій. Розумний лічильник дозволяє правильно обчислити витрати за комунальні послуги та запобігти зайвих витрат для споживача [6].

Для рішень розумного міста використовують телеметрію – це комплекс технологій, з використанням яких можна контролювати різні об'єкти, а також процеси на відстані. Вона дозволяє віддалено збирати та обробляти дані, спостерігати за станом мереж, наприклад, у багатоквартирних та приватних будинках. Також GSM-телеметрія використовується в промисловості, де підприємства та заводи з її допомогою відстежують обсяг споживання ресурсів [6].

Приклад схеми GSM-телеметрії на виробництві наведено на рис. 1.2 [7].

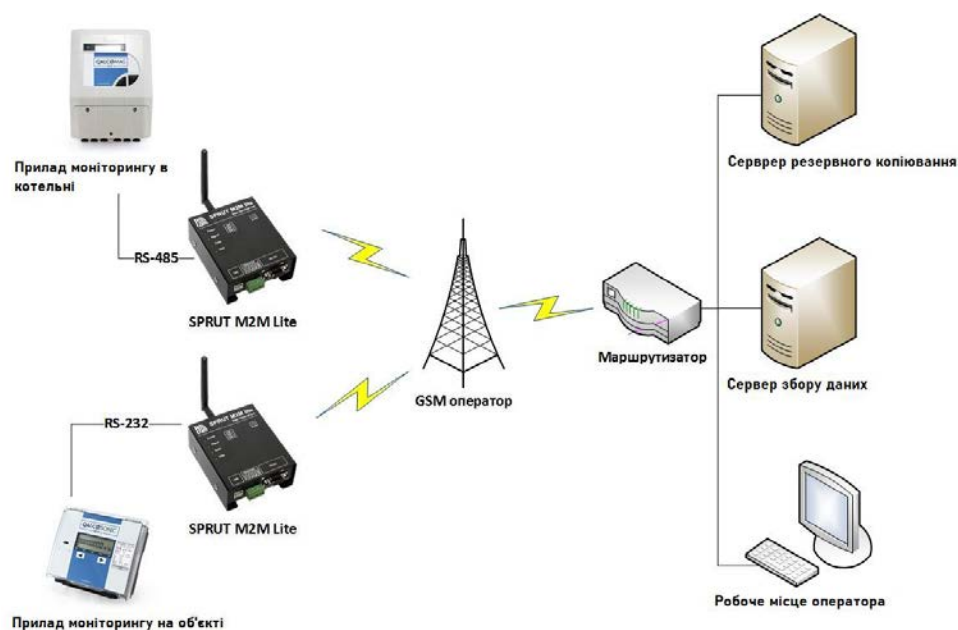


Рисунок 1.2 – Приклад схеми GSM-телеметрії на виробництві [7]

1.2.2 Переваги систем Smart metering

Жителі багатоквартирних будинків отримують низку переваг, використовуючи розумні лічильники, адже якщо датчики встановлені у мережі електроживлення, знижується рівень зайвої витрати за оплату комунальні послуги, адже система Smart metering передає точні показники автоматично, без обов'язкової участі людини [6].

Установка розумних лічильників належить до складу автоматизованої системи обліку, яка збирає та передає показники, серед яких і дані споживання електроенергії. Також система з використанням цих даних будує модель споживання, а також має змогу повідомляти про відхилення. При встановленні датчиків в електромагістралі повідомлення про падіння напруги або обриву лінії приходять до центру управління, в результаті чого про це негайно дізнається аварійна бригада. Завдяки своєчасним повідомленням комунальні служби можуть контролювати стан мереж та мінімізувати витрати від аварій [6].

1.2.3 Принципи роботи систем Smart metering

Мережа розумних датчиків функціонує на основі комунікаційних та інформаційних технологій разом з технологіями Internet of Things (IoT). Застосовуються спеціальне програмне забезпечення (ПЗ), що створене для смарт міста і його компонентів. В телеметрії застосовується стандарт NB-IoT, що використовується приладами з невеликими обсягами обміну інформації. Серед переваг є гнучкість менеджменту використання електроенергії, значна ємність мережі та можливість сигналу проникати глибоко у підвальні приміщення, що містять товстий залізобетонний фундамент [6].

Стійкість до перешкод, а також безпеку та надійність мережевого з'єднання забезпечують технології NB-IoT. Передача інформації використовує

ліцензійні частоти, а тому від втручання зловмисників запобігає державна служба [6].

Технологія може бути реалізована у двох напрямках:

- з новими лічильниками;
- зі старими лічильниками [6].

Приклад лічильника Smart Meters наведено на рис. 1.3 [8].



Рисунок 1.3 – Приклад лічильника Smart Meters [8]

Для використання систем Smart Meters старі лічильники замінюють спеціальним обладнанням, проте перед підключенням до будинку такої конструкції проводять експертне оцінювання та узгоджують спеціальний план. Проте лічильники нового зразку імпульсного типу, що мають інтерфейс wireless M-Bus вимагають тільки під'єднання модему, що має можливість отримувати дані та направляти в хмарну платформу NB-IoT-мережі [6].

Отримані від вузла обліку дані надходять у мережу контролю підприємством окремого каналу електроживлення за допомогою спеціального ПЗ. Центр керування IoT забезпечує віддалене управління SIM-картами і та їх

трафіком. За його допомогою проводяться налаштування віддаленого управління обсягами і частотою передачі інформації постійно та неперервно, а також автоматизується процес управління SIM-картками IoT. Окрім цього, використання особистого кабінету спрощує управління пристроями, їхній віддалений моніторинг, а також надається доступ до аналітики та статистики [6].

Таким чином, в ході проведених досліджень, було встановлено, Smart metering — це система розумних датчиків, для віддаленого обліку та контролю показників лічильників, моніторингу теплоенергії та виявлення аварійних ситуацій. Це також дає можливість скоротити витрати на оплату комунальних послуг. Для цього використовується спеціальне ПЗ, що дозволяє лічильники та датчики у одну мережу.

1.3 Огляд функціональних можливостей існуючих засобів заощадження енергії

1.3.1 Розумна розетка з лічильником енергоспоживання Ajax Socket

Розумна розетка з лічильником енергоспоживання Ajax Socket оснащена захистом від перевантаження, лічильником споживання електроенергії та light-emitting diode (LED) індикацією навантаження. Підключається до системи Ajax відбувається за бездротовим протоколом Jeweller. Відстань зв'язку з централлю Hub становить до 1000 метрів за відсутності перешкод [9].

Socket аналізує, які прилади витрачають найбільше електрики. Як тільки техніка починає споживати енергію, світлодіодне кільце Socket у реальному часі змінює колір підсвічування [9].

Розетка автоматично припиняє подачу живлення на підключені електроприлади під час встановлення чи зняття системи безпеки Ajax з охорони [9].

Споживання енергії у режимі очікування становить менше 1 Вт·год, витримує навантаження до 2,5 кВт. Приклад наведено на рис. 1.4 [9].



Рисунок 1.4 – Розумна розетка Ajax Socket [9]

Містить трирівневий захист побутової техніки від виходу з ладу - відключає подачу живлення при критичному рівні напруги, перевищенні допустимого діапазону струму та/або перегріві та моментально повідомляє про відключення через SMS-повідомлення та push-повідомлення [9].

Серед недоліків є те, що Ajax Socket працює тільки з хабами та ретрансляторами Ajax та керує живленням побутових приладів в програмі Ajax Security System, а тому не може бути модифікованою для системи розумного будинку власної розробки [9].

1.3.2 Розумна WI-FI розетка Sonoff S26

Розумна WI-FI розетка Sonoff S26 є досить популярною серед користувачів подібних девайсів, а тому має під собою безліч прошивок та налаштувань. У даної моделі Sonoff S-26 покращено захист від короткого замикання та високочастотних перешкод загальної електромережі. Містить достатньо високу швидкість відгуку та гарну програмну оптимізацію. Приклад наведено на рис. 1.5 [10].



Рисунок 1.5 – Розумна розетка Sonoff S-26 [10]

Основним інтерфейсом є програма eWeLink, що виглядає як простий будильник, але з розширеним функціоналом. Дана розетка є досить популярною серед користувачів подібних девайсів, а тому має під собою безліч прошивок та налаштувань. Розетка Sonoff S26 є досить вдалим рішенням для інтеграції з системою домашньої автоматизації, адже може бути підключена досить простими способами, а тому дуже якісно використовуватися в рішенні проблеми економії енергії. [10].

1.3.3 Проєкт віддаленого моніторингу енергоефективності та білінгу

Дана система побудована на електронній схемі для вимірювання змінної напруги 230 В та струму в діапазоні 0–5 А підключеного навантаження, розрахунку споживаної енергії та потужності, розрахунку загальної суми на основі різних тарифів і відправлення в хмару та отримання через програму на Android у режимі реального часу. Серед апаратного забезпечення даної системи є МК Arduino Uno, інвазійний датчик струму ACS712 та інвазійний датчик напруги, а також МК ESP8266. Реалізація апаратної схеми показана на рис. 1.6 [11].

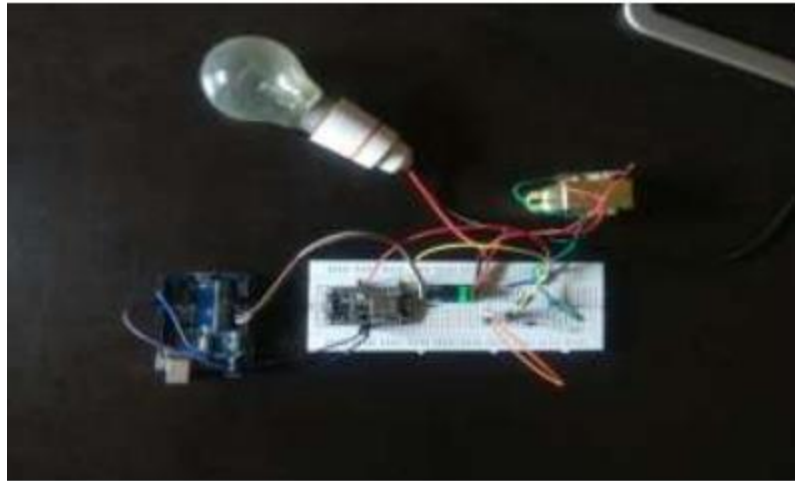


Рисунок 1.6 – Реалізація апаратної схеми [11]

Програмною реалізацією є програмне забезпечення, розроблене інтегрованим середовищем розробки (IDE) для операційної системи Android від компанії Google Android Studio, створене на основі ПЗ JetBrains IntelliJ IDEA та розроблене спеціально для розробки під Android. Показники датчиків струму та напруги системи інтелектуального обліку та виставлення рахунків за енергію показані на рис. 1.7 [11].



Рисунок 1.7 – Програмна реалізація проєкту [11]

Серед недоліків даної системи є необхідність підключення двох МК, використання інвазійних датчиків, а також використання незручного програмного забезпечення.

1.3.4 Розумний лічильник електроенергії з використанням ESP12 та Arduino

Проект розумного лічильника електроенергії на ESP12 та Arduino дозволяє сповістити користувача за допомогою SMS чи електронною поштою, коли кількість споживаної енергії досягне певного значення (межі). Для вимірювання електроенергії використовується датчик струму ACS712. Передача інформації відбувається з використанням модулю ESP8266 за допомогою Wi-Fi та протоколу IFTTT для сповіщень SMS або електронною поштою. Також використовується програмна панель MQTT Android для моніторингу споживання енергії. Зовнішній вигляд проекту розумного лічильника електроенергії наведено на рис. 1.8 [12].

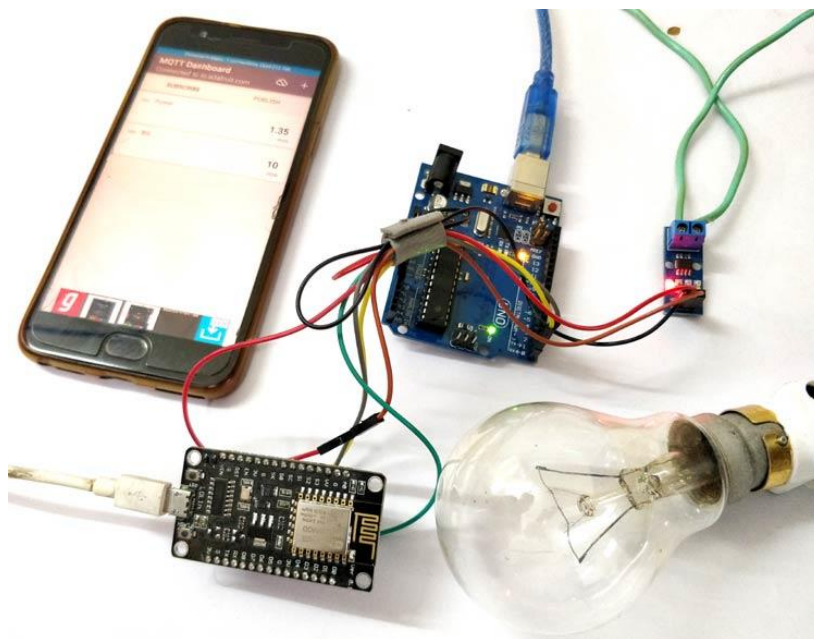


Рисунок 1.8 – Зовнішній вигляд проекту розумного лічильника електроенергії [12]

Таким чином, в проєкті використовуються наступні компоненти:

- плата Arduino Uno;
- ESP12/NodeMCU;
- датчик струму ACS712-30Amp Current sensor;
- будь-який споживач електроенергії змінного струму [12].

У Nodemcu (ESP12) є один аналоговий пін, проте в проєкті він не використовується, адже серія ESP може споживати до 3,3 вольт на своїх пінах. Оскільки в проєкті використовується датчик струму, що на виході може мати до 5 вольт, він може пошкодити модуль Wi-Fi, а тому він не буде використовуватися для зчитувань показників з датчика. Таким чином, Arduino зчитує поточне значення датчика за допомогою аналогового виходу та відправляє його до модуля Wi-Fi ESP12 за допомогою послідовного зв'язку. Принципіальна схема проєкту розумного лічильника електроенергії наведено на рис. 1.9 [12].

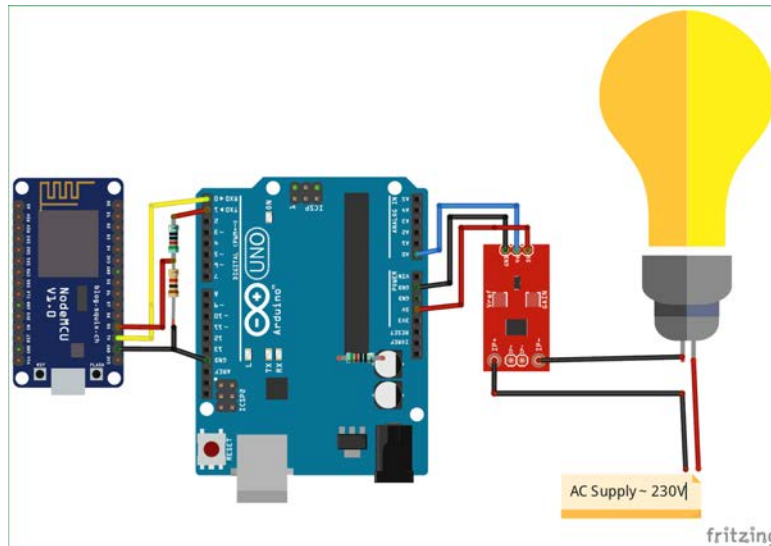


Рисунок 1.9 – Принципіальна схема проєкту розумного лічильника електроенергії [12]

Для моніторингу споживання енергії через Інтернет використовується брокер MQTT. Брокер MQTT використовується як платформа введення-виводу Adafruit. Для зберігання показань лічильника електроенергії

налаштовується обліковий запис Adfruit, а для запуску SMS чи електронної пошти створюється аплет в IFTTT. Приклад відображення результатів моніторингу за допомогою AdaFruit IO наведено на рис. 1.10 [12].

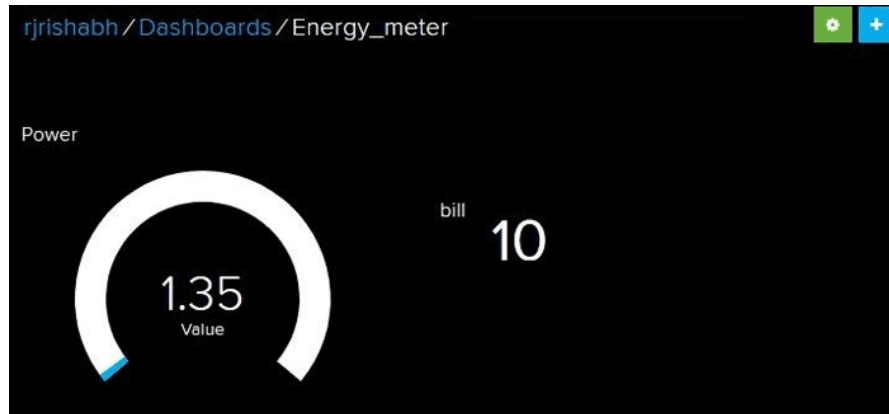


Рисунок 1.10 – Приклад відображення результатів моніторингу за допомогою AdaFruit IO [12]

Також надається можливість відправлення показників лічильника за допомогою SMS чи електронної пошти аплетів IFTTT з використанням аплетів IFTTT. Приклад отримання результатів моніторингу електронною поштою наведено на рис. 1.11 [12].

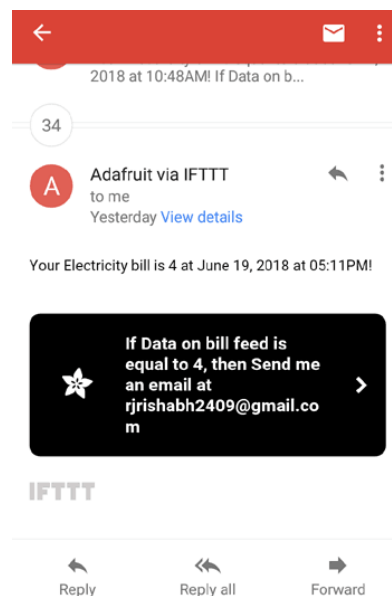


Рисунок 1.11 – Приклад отримання результатів моніторингу електронною поштою [12]

Ще одним способом моніторингу є використання Android-застосунку MQTT Dashboard. Для цього необхідно налаштувати з'єднання з сервером. Приклад отримання результатів моніторингу за допомогою Android-застосунку наведено на рис. 1.12 [12].

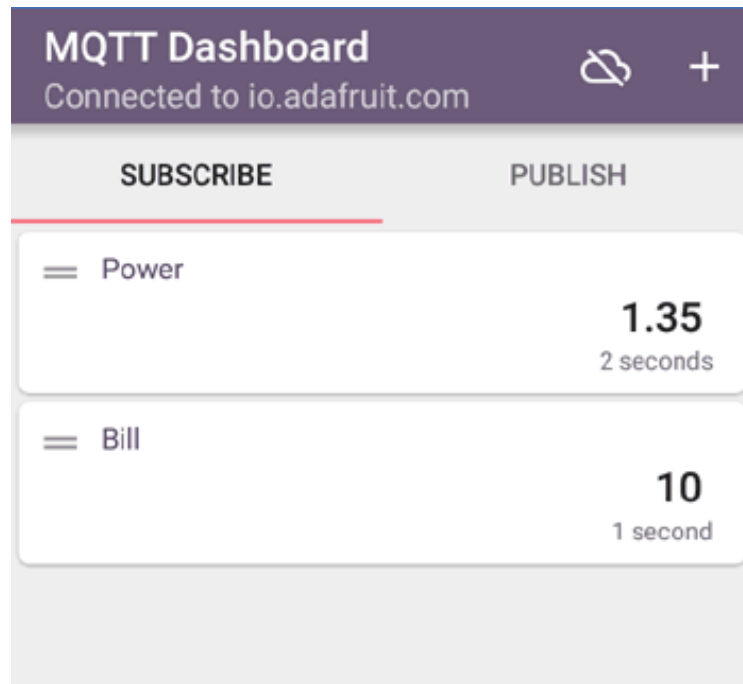


Рисунок 1.12 – Приклад отримання результатів моніторингу за допомогою Android-застосунку [12]

Серед недоліків даної системи є: необхідність підключення двох МК, використання інвазійних датчиків, використання не повного необхідного набору датчиків, а також використання незручного програмного забезпечення.

1.3.5 Інтелектуальний моніторинг споживання електроенергії Raspberry Pi на базі IoT

Даний проєкт розумного лічильника базується на одноплатному комп'ютері Raspberry Pi, що має здатність отримувати дані про витрати електроенергії та завантажувати їх на Adafruit.io. Блок-схема інтелектуального лічильника енергії на базі Raspberry Pi наведено на рис. 1.13 [13].

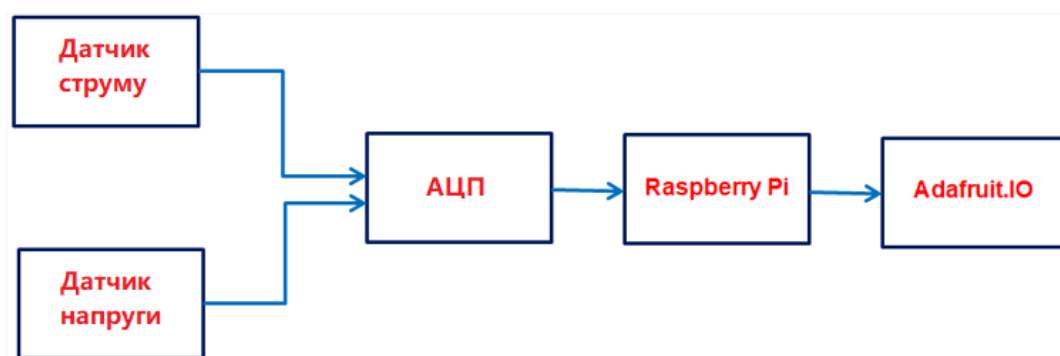


Рисунок 1.13 – Блок-схема інтелектуального лічильника енергії на базі Raspberry Pi [13]

Для збору даного проєкту необхідно використовувати наступні компоненти:

- блок вимірювання струму. Блок вимірювання струму складається з датчика струму SCT-013, який може вимірювати до 100 А. Датчик перетворює струм, що проходить через провід, на якому він закріплений, на менший струм, який потім подається в аналого-цифровий перетворювач (АЦП) через мережу дільників напруги;
- блок вимірювання напруги. В проєкті використовується саморобний безтрансформаторний датчик напруги, який вимірює напругу, використовуючи принцип дільників напруги. Він включає каскад дільника напруги, в якому висока напруга перетворюється на значення, придатне для введення в АЦП;
- блок обробки. Блок обробки складається з АЦП та Raspberry Pi. АЦП приймає аналоговий сигнал і відправляє його на міні-комп'ютер, який потім обчислює точну кількість споживаної потужності та відправляє його у вказану хмару пристроїв. В даному проєкті використовується сервіс Adafruit.io в якості нашої хмари пристроїв [13].

Схеми для проєкту Raspberry Pi Energy Monitor відносно складні і включають в себе підключення блоку датчиків напруги та струму до АЦП, який потім відправляє дані з датчиків на Raspberry Pi. Схему датчика струму наведено на рис. 1.14 [13].

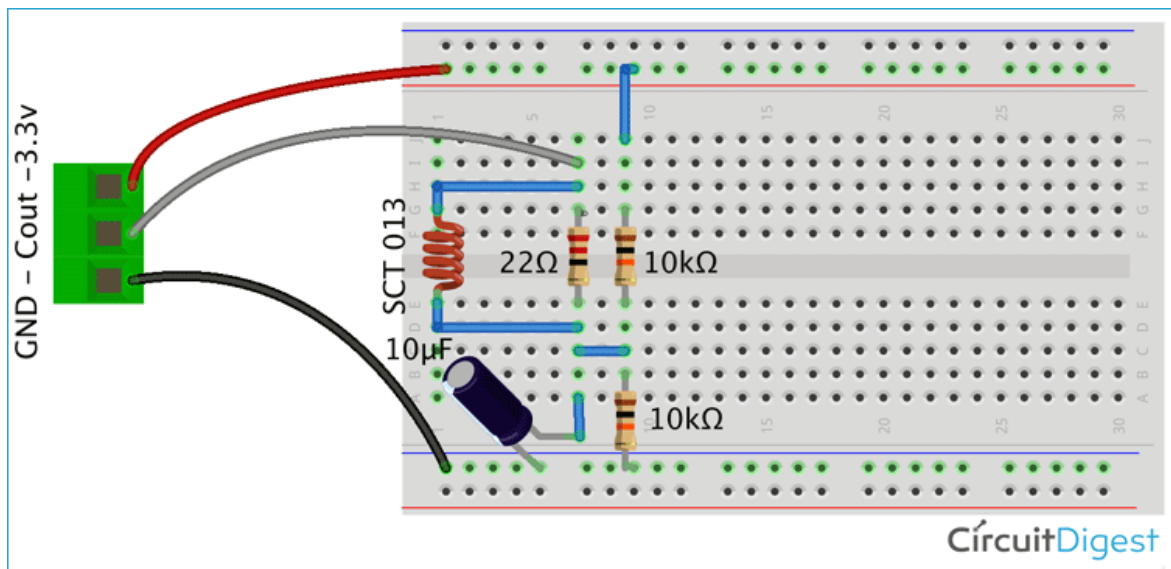


Рисунок 1.14 – Схема датчика струму [13]

Схему датчика напруги наведено на рис. 1.15 [13].

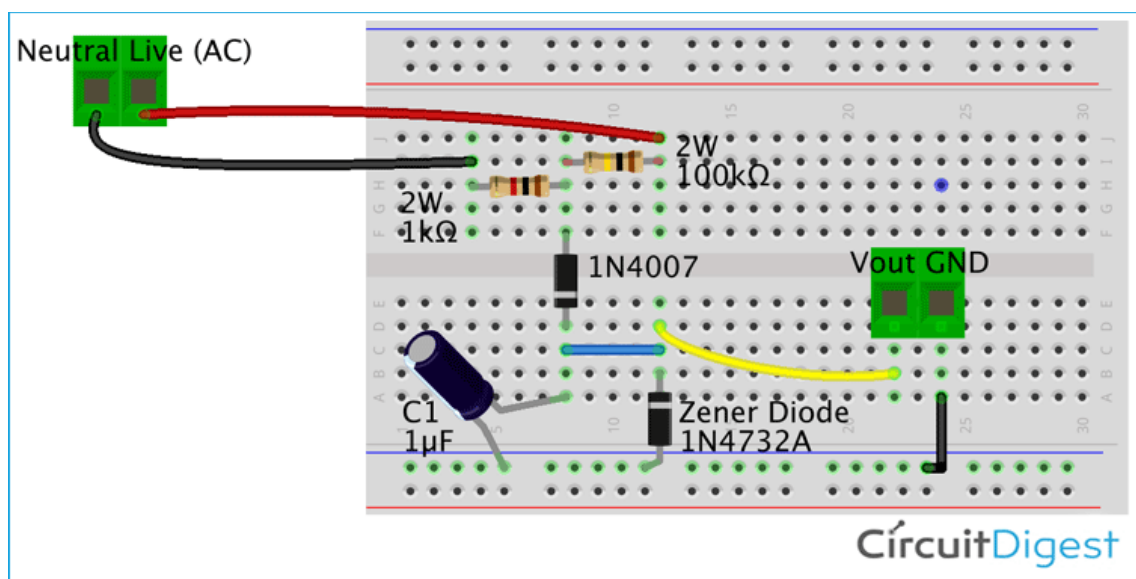


Рисунок 1.15 – Схема датчика напруги [13]

В результаті об'єднується все разом з АЦП (ADS1115), що підключений до raspberry pi, виходів датчиків струму та напруги, підключених до контактів A0 та A1 ADS1115 відповідно. Схему процесорного блоку наведено на рис. 1.16 [13].

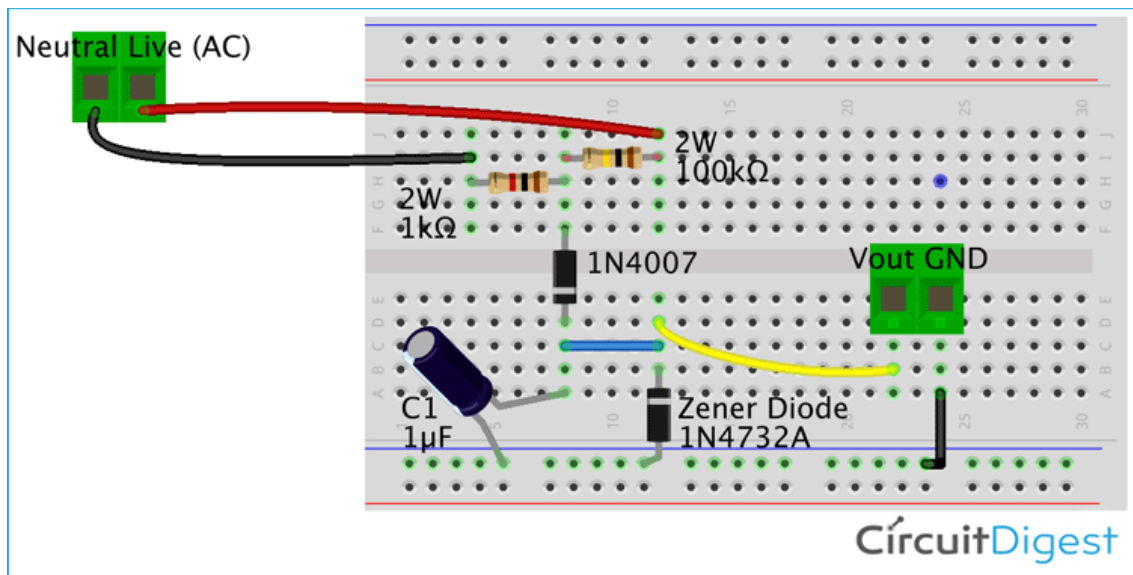


Рисунок 1.16 – Схема процесорного блоку [13]

Таким чином, в результаті фінальної збірки компонентів зовнішній вигляд проєкту буде як наведено на рис. 1.17 [13].

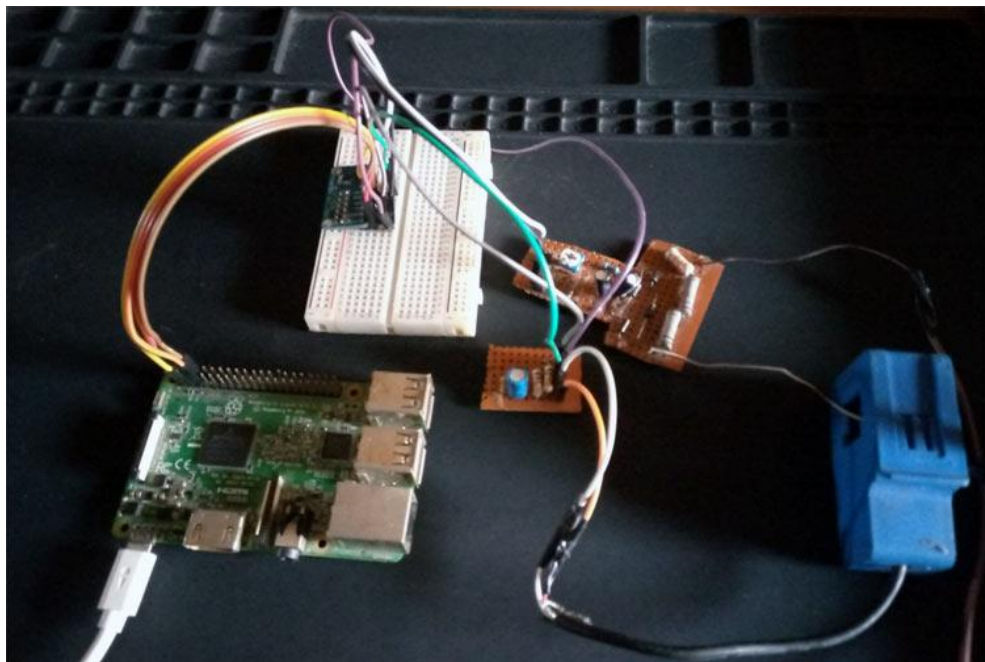


Рисунок 1.17 – Зовнішній вигляд проєкту [13]

Після фінального складання розумний лічильник підключається до панелі моніторингу на adafruit.io, де буде відображатися компонент графіку

для можливості отримання графічного представлення даних, як показано на рис. 1.18 [13].

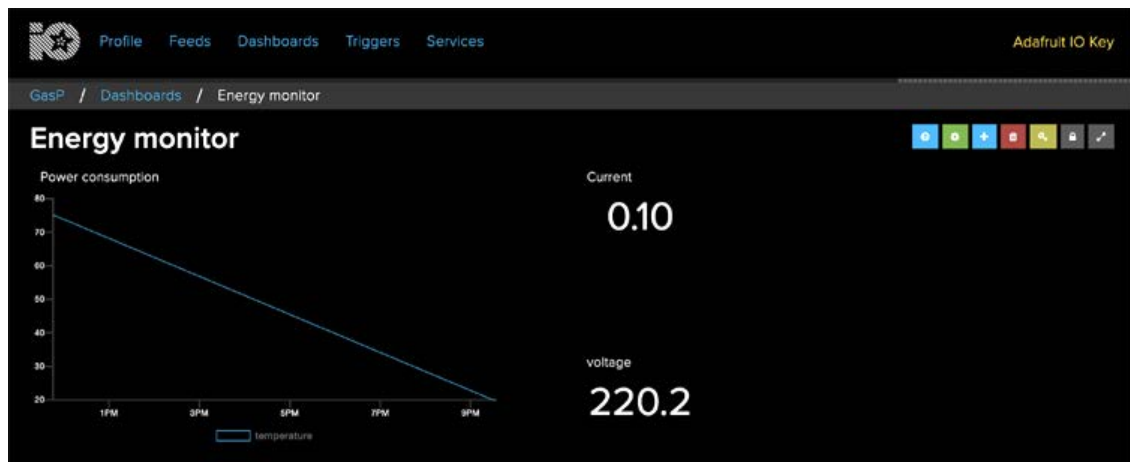


Рисунок 1.18 – Приклад графічного представлення результатів моніторингу [13]

Серед недоліків даної системи є необхідність підключення міні-комп'ютера raspberry pi, використання саморобних датчиків, складна схема підключення, а також використання незручного програмного забезпечення.

1.3.6 Система моніторингу енергоефективності PEEMS

Система моніторингу PEEMS включає батареї і однофазний інвертор напруги з цифровим управлінням, яким можна керувати як джерелом струму або джерелом напруги в залежності від стану мережі змінного струму і налаштувань користувача [14].

PEEMS гарантує подачу живлення на критичні навантаження при виході з експлуатації мережі змінного струму, у цьому випадку однофазний інвертор напруги керується як джерело напруги. Він також забезпечує контроль пікової потужності, подаючи живлення від батареї на локальні навантаження, поки вони живляться від мережі змінного струму, якщо навантаження стає більшим. Оцінюється економія витрат за електроенергію з

допомогою скорочення пікових навантажень. Використовується схема енергоефективності на основі МК, яка може бути контролером живлення та показувати загальне навантаження і скільки енергії використовується день у день [14].

Використовується середньоквадратичне значення та трансформатор струму з використанням та підрахунком середнього значення для навантаження, що є простим способом обчислити потужність. В схемі використовується МК Atmega328, мосфет Mosfet-IRZ-44, резистор номіналом 100k та рідкокристалічний екран 16x2. Частота живлення 16 МГц [14].

Щоб забезпечити живленням критичні навантаження при збої мережі змінного струму, система виявляє відмову мережі та діє як джерело напруги для критичних навантажень. У цьому режимі порушення форми хвилі напруги стає помітним, коли система управління електроспоживанням повторно підключає мережу змінного струму до навантажень. У діодному випрямлячі також присутній пусковий струм, тому що постійна напруга випрямляча дещо просідає під час дії ізолюваного режиму. При цьому змінна напруга, створювана системою управління електроспоживанням в ізолюваному режимі, трохи менша (близько 7 %), ніж змінна напруга мережі [14].

Дана система має ряд переваг, наприклад ця конструкція дуже корисна для електричного обслуговування силового агрегату, наявна система захисту від великих втрат для трансформатора, наявний контроль за навантаженням, блок розподілу живлення виконаний простим способом, а також дизайн системи має низьку вартість [14].

1.3.7 Переваги і недоліки наявних систем заощадження енергії

Таким чином, в процесі аналізу наявних систем моніторингу і заощадження електроенергії було виокремлено ряд наявних переваг і недоліків, що наведено в табл. 1.1.

Таблиця 1.1 – Порівняльна таблиця наявних систем віддаленого моніторингу і контролю енергоефективності

Назва	Переваги	Недоліки
1	2	3
Розумна розетка з лічильником енергоспоживання Ajax Socket	Можливість відправки SMS-повідомлення та push-повідомлення, автоматично припиняє подачу живлення, наявність бездротового протоколу Jeweller	Працює тільки з хабами та ретрансляторами Ajax, керує живленням лише в програмі Ajax Security System, не може бути модифікованою для системи розумного будинку власної розробки
Розумна Wi-Fi розетка Sonoff S26	Wi-Fi інтерфейс, дистанційне керування, 5 режимів таймера, не вимагає навичок програмування	Не може бути модифікованою для системи розумного будинку власної розробки, не має можливості моніторингу потужності
Проект віддаленого моніторингу енергоефективності і білінгу	Відправлення даних в хмару, отримання результатів через програму на Android у режимі реального часу	Необхідність підключення двох МК, використання інвазійних датчиків, незручне програмне забезпечення
Розумний лічильник електроенергії з використанням ESP12 та Arduino	Сповіщення за допомогою SMS чи електронної пошти, використання брокера MQTT, керування в платформі введення-виводу Adafruit	Необхідність підключення двох МК, використання інвазійних датчиків і незручного програмного забезпечення
Інтелектуальний моніторинг споживання електроенергії Raspberry Pi на базі IoT	Можливість графічного представлення даних, підключається до панелі моніторингу на adafruit.io, вимірювання струму до 100 А	Необхідність підключення міні-комп'ютера raspberry pi, використання саморобних датчиків, складна схема підключення,

Продовження таблиці 1.1

1	2	3
Система моніторингу енергоефективності PEEMS	Наявна система захисту від втрат для трансформатора, наявний контроль за навантаженням, блок розподілу живлення виконаний простим способом	Немає бездротової передачі результатів моніторингу

В ході проведення дослідження функціональних можливостей існуючих засобів заощадження енергії було встановлено, що на сьогоднішній день наявні системи мають свої переваги та недоліки.

1.4 Постановка задач дипломної роботи

Таким чином, в ході досліджень, було проаналізовано системи домашньої автоматизації та системи контролю енергозбереження в розумному будинку. Також показано наявні програмно-апаратні засоби та досліджено їх функціональні особливості.

В результаті було встановлено, що в своїх роботах автори пропонують різні варіанти для вирішення задачі.

В проєктах розумної розетки Ajax Socket та Sonoff S26 виконується бездротова передача даних, проте вони є дещо незручними, адже працюють лише в своїх екосистемах, тому не можуть бути зручно інтегрованими до систем розумного будинку.

Проєкт віддаленого моніторингу енергоефективності і білінгу надає можливість спостерігати за результатами в застосунку на Android, а розумний лічильник електроенергії з використанням ESP12 та Arduino – за допомогою SMS чи електронної пошти та за допомогою платформи Adafruit. Проте обидва

варіанти використовують інвазійні датчики та незручного програмного забезпечення, а також вимагають підключення двох МК.

Інтелектуальний моніторинг споживання електроенергії Raspberry Pi на базі IoT Також надає можливість виводити дані за допомогою платформи Adafruit, проте вимагає недоцільне підключення міні-комп'ютера Raspberry Pi, а також не є досить точним.

Система моніторингу енергоефективності PEEMS не має бездротової передачі результатів моніторингу, тобто виводить дані лише локально.

Загалом, системи віддаленого моніторингу контролю електроспоживання є сучасною актуальною проблематикою, що потребує вирішення, а тому тема даної дипломної кваліфікаційної роботи є актуальною.

Мета роботи – дослідження та розробка системи віддаленого моніторингу споживання електроенергії для контролю процесу енергоспоживання, підвищення ефективності споживання електроенергії та безпеки при використанні електроприладів.

Задачі роботи:

- дослідження та вибір технологій та засобів прототипування системи віддаленого моніторингу споживання електроенергії;
- розробка та тестування апаратних та програмних засобів системи;
- розробка програмної документації.

2 ДОСЛІДЖЕННЯ ТА ВИБІР ТЕХНОЛОГІЙ ТА ЗАСОБІВ ПРОТОТИПУВАННЯ

2.1 Програмні засоби підтримки процесу прототипування

Загалом, для розробки проєктів на основі МК Arduino UNO, ESP8266 та ESP32, які використовувалися в ході виконання роботи, можна використовувати як написання коду програми, так і візуальне програмування.

Яскравим прикладом візуального програмування є Node-RED.

Node-RED – це інструмент програмування для об'єднання апаратних пристроїв, application programming interface (API) та онлайн-сервісів новими та цікавими способами. Він надає редактор на основі браузера, який дозволяє легко об'єднувати потоки, використовуючи широкий діапазон вузлів на панелі, які можна розгорнути серед виконання одним клацанням миші. Приклад використання Node-RED наведено на рис. 2.1 [15].

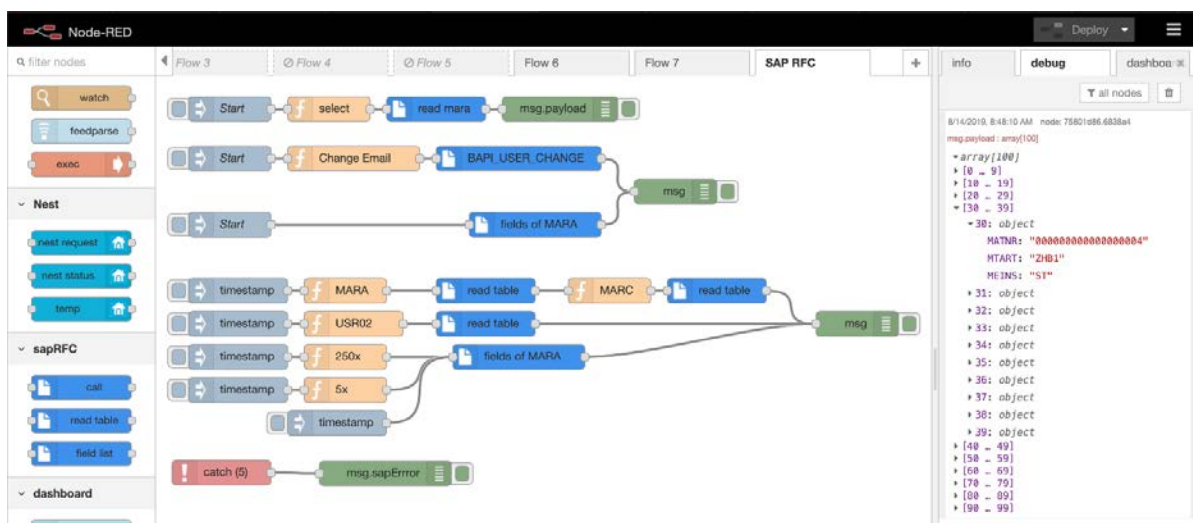


Рисунок 2.1 – Приклад використання Node-RED [15]

Node-RED представляє редактор потоків на основі браузера, який дозволяє легко зв'язати потоки разом, використовуючи широкий спектр вузлів в палітрі. Далі потоки можна досить просто розвернути в середовищі виконання. Функції JavaScript можна створювати в редакторі за допомогою

редактора форматованого тексту. Вбудована бібліотека дозволяє зберігати корисні функції, шаблони або потоки для повторного використання [15].

Проте засобів для кодування програм набагато більше. Досить простим та зручним використанням вважається середовище розробки Arduino IDE. Існує 2 варіанти:

- при наявності стабільного підключення до мережі Internet, можна використовувати онлайн-середовище Arduino Web Editor. З його допомогою можна зберігати проєкти в хмарному сховищі, що надає доступ до них з будь-якого пристрою, а також це дозволяє зберігати резервні копії. Таким чином буде усунена необхідність постійного оновлення версії, а також власне встановлення середовища безпосередньо на комп'ютер;
- для роботи в автономному режимі надається версія IDE, що встановлюється на комп'ютер. З її використанням відпадає необхідність постійного підключення до мережі Internet для роботи з МК [16].

Приклад інтерфейсу програмування в Arduino IDE наведено на рис. 2.2 [16].

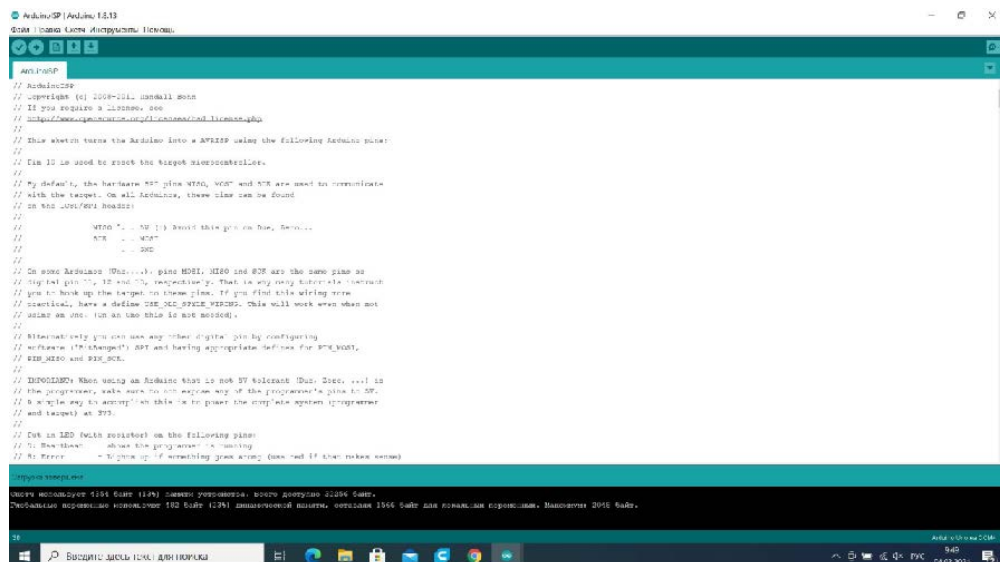


Рисунок 2.2 – Приклад інтерфейсу програмування в Arduino IDE [16]

Середовище розробки Arduino IDE містить вбудований редактор тексту для написання коду, вікно повідомлень, спеціально область для виведення

тексту – консоль, а також панель інструментів і меню. Для завантаження програм, а також для зв'язку, необхідне підключення до апаратної частини Arduino [16].

Окрім стандартного вбудованого редактору коду можна використовувати також і сторонні, наприклад Notepad++ чи Visual Studio Code. Останній також окремо може надавати за допомогою сторонніх розширень, таких як PlatformIO, власне середовище розробки, яке можна використовувати як повноцінний застосунок для програмування МК. Приклад його використання зображено на рис. 2.3 [17].

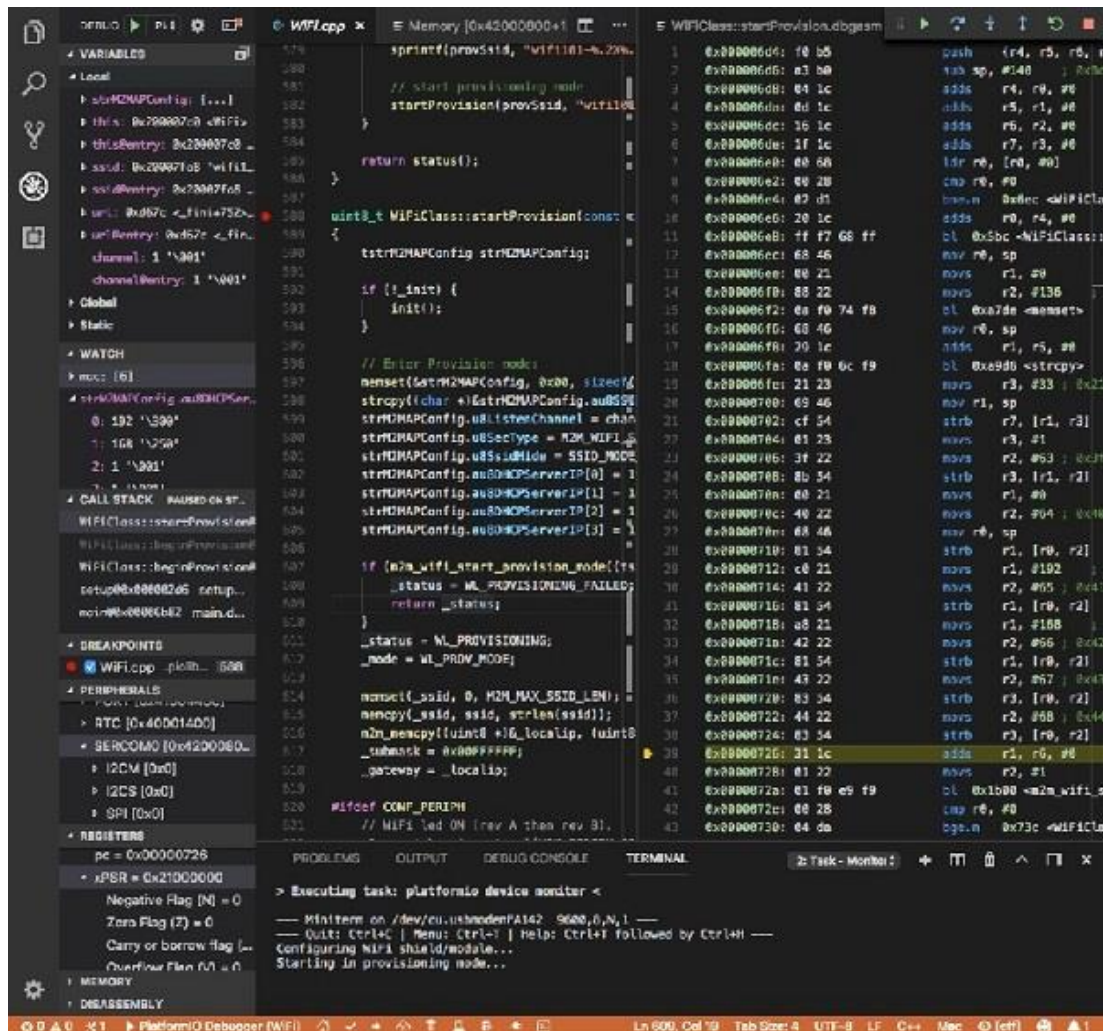


Рисунок 2.3 – Приклад інтерфейсу програмування коду в PlatformIO [17]

Також важливо зазначити, що, окрім написання мовою C++, за допомогою певних засобів можна програмувати також і мовою Python [18].

Оскільки МК Arduino є частиною сімейства AVR, то для їх програмування можна використовувати й таке програмне забезпечення, як Atmel Studio. Це інтегроване середовище розробки від компанії Atmel для розробки проєктів під МК ARM Cortex-M і AVR. Дана програма дозволяє писати код мовою асемблера та на C/C++. Приклад використання Atmel Studio зображено на рис. 2.4 [19].

Окремо необхідно зазначити, що МК ESP32 та ESP8266 можна програмувати мовою Lua, а також як і Arduino мовою Python. Для цього можна наприклад використовувати веб-IDE ChiliPerpr. Приклад програмування мовою Lua в веб-IDE ChiliPerpr зображено на рис. 2.5 [20].

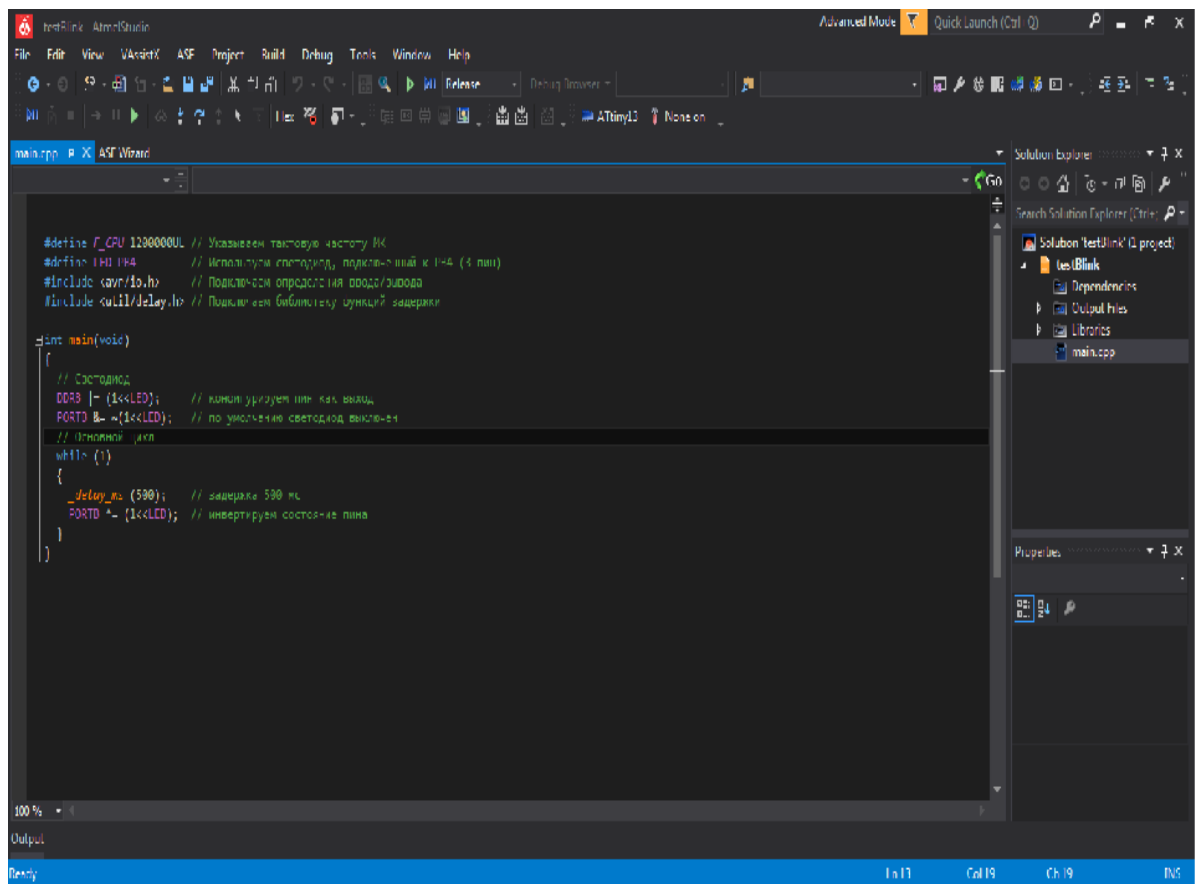


Рисунок 2.4 – Приклад використання Atmel Studio [19]

```

1  -- ChiliPeppr - Blink built-in LEDs on ESP32 devices
2  m = {}
3
4  -- Config
5  -- Try pins 2 on DOIT, 13 on TTGO, 16 on Wemos
6  m.pins = {ESP32_DOIT = 2, ESP32_Lolin32 = 5, ESP32_TTGO = 13, ESP32_Wemos = 16}
7  m.value = 0
8  m.duration = 500
9  m.ctr = 0
10
11 function m.toggleLED ()
12   if m.value == 0 then
13     m.value = 1
14   else
15     m.value = 0
16   end
17
18   -- print("Toggling")
19   for key, value in pairs(m.pins) do
20     local pin = value
21     gpio.write(pin, m.value)
22     -- print("Toggled " .. pin)
23   end
24
25   m.ctr = m.ctr + 1

```

Рисунок 2.5 – Приклад програмування мовою Lua в веб-IDE ChiliPeppr [20]

Таким чином, в процесі аналізу наявних середовищ програмування було виокремлено ряд наявних переваг і недоліків, що наведено в табл. 2.1.

Таблиця 2.1 – Порівняльна таблиця наявних середовищ програмування

Назва	Переваги	Недоліки
1	2	3
Node-RED	Написання скриптів мовою JavaScript, програмування в браузері, наявність візуального програмування	Незручність написання складних проєктів, відсутність багатьох бібліотек
Arduino IDE	Наявність великої кількості бібліотек, відносна простота написання коду, зручний вбудований плотер графіків	Відсутність сторонніх плагінів, відсутність певних функцій сучасних редакторів коду
Visual Studio Code	Наявність великої кількості сторонніх плагінів, зручність роботи з кодом, зручний файловий менеджер	Незручне налаштування перед написанням коду, відсутність вбудованого плотера графіків, погана підтримка бібліотек

Продовження таблиці 2.1

1	2	3
Atmel Studio	Можливість писати код мовою асемблера та на C/C++, велике різноманіття підтримки МК	Висока складність, відсутність певних бібліотек
ChiliPeppr	Можливість програмування ESP32 та ESP8266 мовою Lua	Високий поріг входу, низька популярність

Для моніторингу показників було досліджено декілька варіантів.

Один з варіантів – це використання можливостей МК esp32 та esp8266. Однією з функцій, яку забезпечують МК, є в тому, що можуть не тільки підключатися до існуючої Wi-Fi мережі та працювати як веб-сервер, але і встановлювати власну мережу, дозволяючи іншим пристроям підключатися безпосередньо до нього та отримувати доступ до веб-сторінок. Це можливо, адже МК може працювати в трьох різних режимах: режим станції, режим точки доступу та обидва перші режими одночасно. Схеми режимів наведено на рис. 2.6 та 2.7 [21].

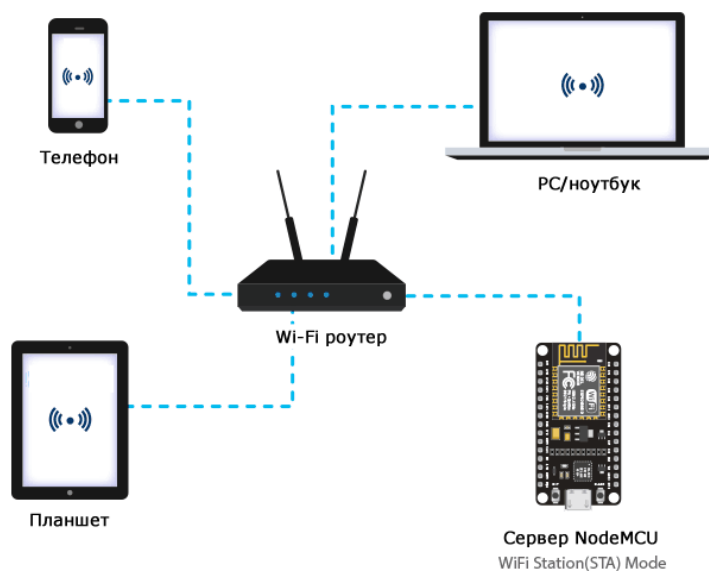


Рисунок 2.6 – Режим станції (STA) [21]

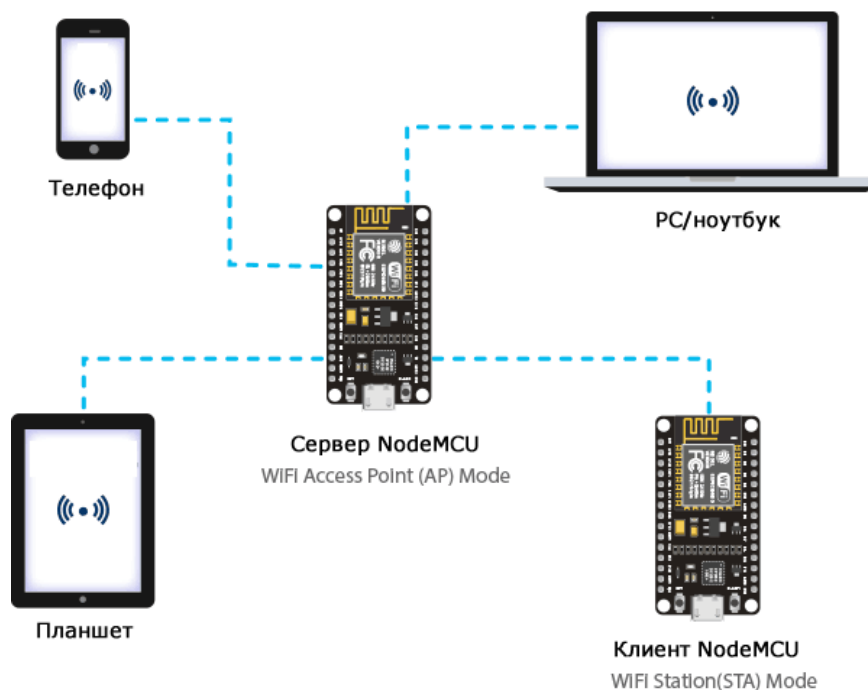


Рисунок 2.7 – Режим точки доступа (AP) [21]

В результаті надається можливість моніторити систему, а також керувати за допомогою браузера, використовуючи підключення по IP-адресу. Приклад такого підключення наведено на рис. 2.8 [21].

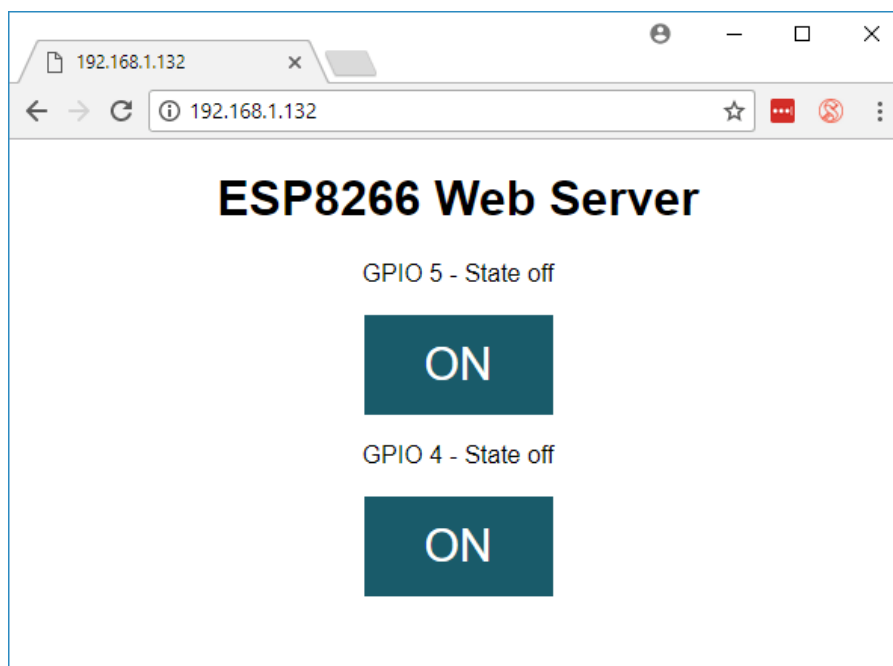


Рисунок 2.8 – Сторінка моніторингу системи [21]

Даний варіант має певні обмеження, адже даний варіант не передбачає інтегрування з доступними системами віддаленого моніторингу через те, що вимагає підключення до МК, знаходячись у одній локальній мережі.

Ці обмеження можна уникнути шляхом перекиданням портів на роутері або використання тунелів, наприклад за допомогою Secure Shell (SSH) чи ngrok [22].

Варто зазначити, що МК esp8266 та esp32 також можуть керуватися та моніторитися напрямку через сервіс Телеграм. Приклад використання наведено на рис. 2.9 [23].

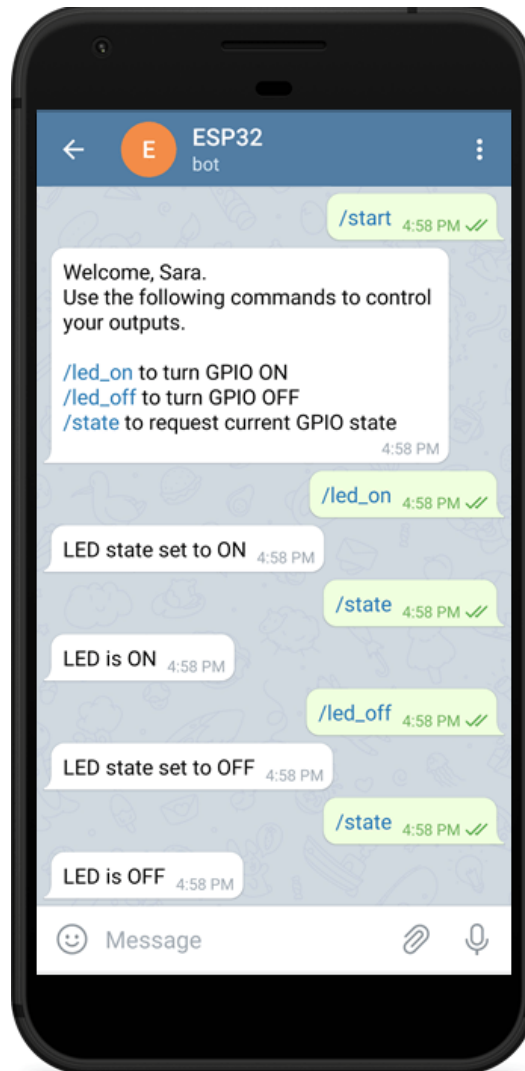


Рисунок 2.9 – Приклад використання Телеграм для моніторингу МК [23]

Досить непоганим засобом моніторингу є використання сервісу Thinger.io, який дозволяє підключати свої продукти Інтернету речей, а також дає змогу керувати ними. Приклад інтерфейсу наведено на рис. 2.10 [24].

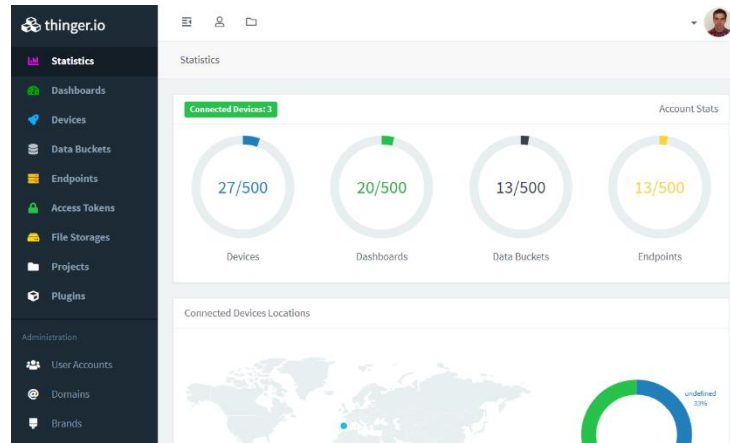


Рисунок 2.10 – Панель моніторингу системи Thinger.io [24]

Досить раціональним рішенням є використання openHAB. Відкрита шина домашньої автоматизації openHAB – це платформа автоматизації дому з відкритим кодом, яка не залежить від технологій, та працює як центр розумного будинку. Тому openHAB є системою домашньої автоматизації, що дозволяє за допомогою MQTT підключати віддалені МК, керувати ними та моніторити їхній стан, що дозволяє спостерігати за енергоспоживанням. Приклад панелі openHAB наведено на рис. 2.11 [25].

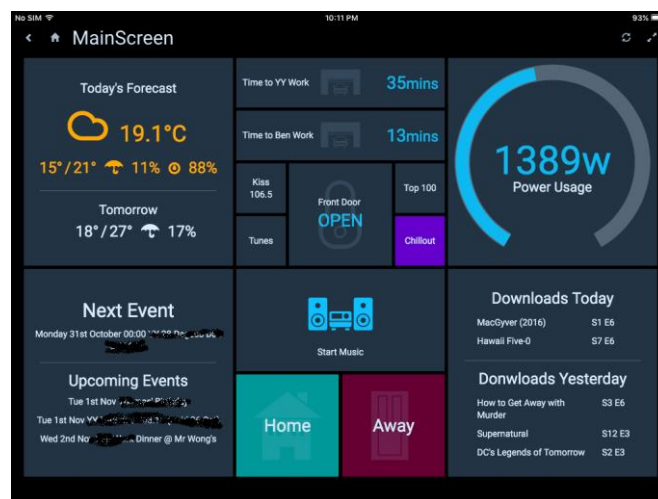


Рисунок 2.11 – Приклад панелі openHAB [25]

Таким чином, в процесі аналізу наявних способів моніторингу та керування системи контролю енергоефективності було виокремлено ряд наявних переваг і недоліків, що наведено в табл. 2.2.

Таблиця 2.2 – Порівняльна таблиця наявних способів моніторингу та керування системи контролю енергоефективності

Назва	Переваги	Недоліки
Веб-сторінка	Високий рівень кастомізації інтерфейсу, унікальність дизайну, можливість локального використання	Необхідність створювати інтерфейс з нуля, відсутність готових інструментів для створення інтерфейсу
Телеграм	Зручність інтеграції з популярним месенджером, відносна простота	Можливість взаємодії лише текстовими командами, низька наочність результатів моніторингу, неможливість локального використання
Thingier.io	Наявність готового хмарного сервісу, наявність готових компонентів інтерфейсу	Погана інтеграція з багатьма компонентами IoT, неможливість локального використання
Панель OpenHAB	Наявність готових компонентів інтерфейсу, гарна інтеграція з багатьма компонентами IoT, можливість локального використання	Погана документація, неможливість повної кастомізації інтерфейсу

В системі домашньої автоматизації між пристроями має бути налаштований зв'язок передачі даних, за допомогою якого МК мають

здатність отримувати дані з датчиків, обробляти її та передавати до центральної системи. Комунікація відбувається за певним протоколом. Таким чином необхідно дослідити наявні протоколи та виокремити їхні особливості [26].

Bluetooth і Bluetooth Low Energy (BLE) — це бездротова технологія малого радіусу дії, яка використовує короткохвильові ультрависокочастотні радіохвилі. Найчастіше він використовується для потокового аудіо, але також є і засобом для бездротових пристроїв. Тому цей варіант підключення з низьким енергоспоживанням і низьким радіусом дії є ідеальним як для персональних мереж, так і для розгортання Інтернету речей. Іншим варіантом є BLE, який є новою версією, оптимізованою для з'єднань IoT, який споживає менше енергії, ніж стандартний Bluetooth, а тому також часто використовується [27].

Стільниковий зв'язок є одним із досить поширених і відомих варіантів, доступних для додатків Інтернету речей, і це один із найкращих варіантів для розгортання, де зв'язок здійснюється на більших відстанях. Незважаючи на те, що застарілі стандарти стільникового зв'язку 2G і 3G зараз поступово припиняються, телекомунікаційні компанії швидко розширюють охоплення нових високошвидкісних стандартів, а саме 4G [28].

Протокол MQTT використовує гнучкий механізм маршрутизації та асинхронний зв'язок поверх стека TCP/IP, заснований на парадигмі публікації-підписки. MQTT складається з трьох основних компонентів: передплатника, видавця та брокера. Він розроблений для роботи в ситуаціях з низькою пропускнуою здатністю, наприклад для датчиків і мобільних пристроїв у ненадійних мережах. Ця здатність робить його найкращим варіантом для підключення пристроїв із невеликим кодом, а також для бездротових мереж із різними рівнями затримки, що виникають через обмеження пропускнуої здатності або ненадійні з'єднання. Таким чином, наразі MQTT є одним із найголовніших протоколів з відкритим кодом для підключення IoT до промислових пристроїв IoT [29].

HTTP є найпоширенішим способом передачі ресурсів у Інтернеті, що визначає формат та призначення повідомлень, якими обмінюються Web-компоненти, такі як клієнти та сервери. Та спосіб інтерпретації полів кожного рядка повідомлення. У середовищі IoT звичайне використання HTTP полягає в тому, щоб дозволити пристроям виконувати запити POST до ресурсу, який представляє стан пристрою в службі IoT [30].

WebSocket дозволяє підтримувати постійні з'єднання з одночасним двонаправленим зв'язком. Він має усі переваги HTTP, оскільки веб-сокети спочатку починаються як рукописання HTTP, а потім підвищуються для продовження іншої частини зв'язку у веб-сокетах. Основа веб-сокетів полягає в стратегії на основі push. Це означає, що на відміну від опитування або довгострокового опитування, яке передбачає регулярні запити сервера про наявність нових даних, push-сервер сам надсилає повідомлення зацікавленому клієнту щоразу, коли стають доступними нові дані, що називається парадигмою публікації-підписки або просто pub/sub. Дана технологія є можливою, оскільки веб-сокети зберігають з'єднання відкритим, доки його не закрий одна з двох сторін, тобто клієнт або сервер [31].

В ході аналізу було виокремлено особливості протоколів даних. Результати наведено в табл. 2.3.

Таблиця 2.3 – Порівняльна таблиця особливостей протоколів передачі даних.

Протокол	Особливості
1	2
WebSocket	Великий рівень енергоспоживання, відправка даних вище за HTTP, можливість роботи в асинхронному режимі, можливий розрив з'єднання
Стільниковий зв'язок	Необхідність використання додаткового обладнання, можливість відправлення великих об'ємів даних, більш висока вартість використання, велике енергоспоживання

Продовження таблиці 2.3

1	2
Bluetooth / BLE	Низький рівень енергоспоживання, малий радіус дії
MQTT	Малі затримки, низький рівень енергоспоживання, можливість роботи в умовах з низькою пропускнуою здатністю
HTTP	Можливі затримки передачі даних, можуть бути проблеми з цілісністю та конфіденційністю даних, є можливість обробляти великі об'єми даних

В ході аналізу було виявлено, що для цілей забезпечення надійності передачі даних, безпеки та зниження енергоспоживання для передачі результатів вимірювання показників використання електроенергії до системи домашньої автоматизації найдоцільніше використати протокол передачі даних MQTT.

Таким чином, в ході аналізу програмних засобів підтримки процесу прототипування було обрано в якості середовища програмування Arduino IDE, написання коду мовою C++ та в якості системи моніторингу використовується система домашньої автоматизації OpenHAB з використанням підключення по MQTT.

2.2 Апаратні засоби підтримки процесу прототипування

2.2.1 Передмова

Для проведення аналізів та тестування, а також прототипування було проаналізовано ряд апаратних засобів.

До апаратних засобів належать МК, датчики, розбірна розетка, а також через неможливість виконання замірів дійсного значення сили змінного струму у зв'язку з виходом з ладу вимірювального приладу на час тестувань,

було використано деякі електроприлади з маркуванням, на якому були вказані номінальні параметри.

2.2.2 Параметри приладів для тестування

В якості приладів для тестування було відібрано 2 електроприлади з потужністю 2000 Вт та 1800 Вт з силою струму відповідно 9.1 А та 8.2 А.

Параметри вказаних електроприладів наведено на рис. 2.12 та 2.13.



Рисунок 2.12 – Приклад параметрів електроприладу для тестування



Рисунок 2.13 – Приклад параметрів використаного електроприладу для тестування

Таким чином було обрано прилади для тестування системи, які наведено в табл. 2.4.

Таблиця 2.4 – Таблиця параметрів приладів для тестування системи

Назва	Максимальний струм	Максимальна потужність
Будівельний фен	2000 Вт	9.1 А
Побутовий фен	1800 Вт	8.2 А

2.2.3 Параметри мікроконтролерів

В якості МК в ході досліджень використовувалися Arduino UNO, ESP8266 та ESP32.

Arduino Uno – плата МК на базі Atmega328P. Вона має 14 цифрових вхідних/вихідних контактів, з яких 6 можуть використовуватися як виходи широтно імпульсної модуляції (ШИМ), 6 аналогових входів, керамічний резонатор 16 МГц, USB-з'єднання, роз'єм живлення, роз'єм ICSP та кнопку скидання. Містить все необхідне для підтримки МК. Приклад плати Arduino Uno наведено на рис. 2.14 [32].



Рисунок 2.14 – Приклад плати Arduino Uno [32]

ESP8266 можна використовувати як зовнішній модуль Wi-Fi, використовуючи стандартне мікропрограмне забезпечення набору AT Command, підключивши його до будь-якого МК за допомогою послідовного UART, або безпосередньо використовувати як МК з підтримкою Wi-Fi,

запрограмувавши нову мікропрограму за допомогою наданого SDK. Виводи GPIO забезпечують аналоговий і цифровий введення, а також ШІМ, SPI, I2C тощо. Плата використовується в основному в контексті IoT, коли необхідно додати підключення, наприклад, до проєкту Arduino. Проте доволі часто виступає і основою проєкту. Приклад плати ESP8266 наведено на рис. 2.15 [33].

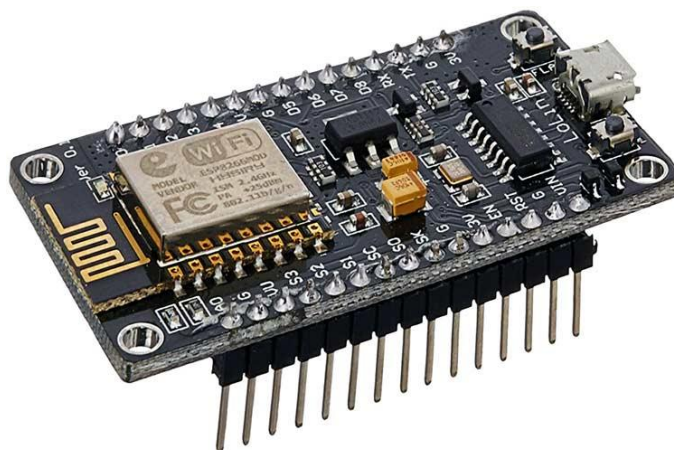


Рисунок 2.15 – Приклад плати ESP8266 [33]

Модуль розробника LuaNode32 побудований на мікромодулі ESP-WROOM-32 – новому мініатюрному високопродуктивному чіпі Wi-Fi + BT + BLE від компанії Espressif, призначеному для широкого спектру застосувань, починаючи від малопотужних мережевих датчиків до найскладніших додатків, наприклад, таких як кодування, потокова передача музики та MP3 кодування. На модулі зібрано всю необхідну мінімальну периферію, достатню для швидкого та комфортного старту роботи з ESP-WROOM-32 [34].

ESP-WROOM-32 виконаний на базі популярного двоядерного чіпсету ESP32, зі змінною тактовою частотою від 80 МГц до 240 МГц, можливістю індивідуального керування та живлення [34].

Модуль розроблений для переносної та автономної електроніки та додатків інтернет-речей, виконаний у мініатюрному корпусі 25,5 мм x 18 мм, має на борту Flash пам'ять, кварц 40 МГц та PCB антену, що забезпечує відмінні RF характеристики [34].

ESP-WROOM-32 має багату периферію, що включає такі інтерфейси як UART, SPI, I²C, I²S, роз'єм для SD карти, інфрачервоний порт, інтерфейс для підключення ємнісної сенсорної панелі. Однією з особливостей модуля є наднизьке споживання і гнучкий вибір режимів, що «сплять», що дозволяють отримати цифри до 20мкА (deep sleep mode). Модуль підтримує весь стек протоколів стандартів WiFi 802.11n та BT4.2, забезпечуючи цей функціонал через інтерфейси SPI/SDIO або I²C/UART. Приклад плати ESP32 наведено на рис. 2.16 [34].



Рисунок 2.16 – Приклад плати ESP32 [34]

M5Stack Atom Lite є досить компактною платою для розробки з серії комплектів для розробки M5Stack. Він надає більше GPIO для налаштування користувача, що підходить для вбудованих пристроїв розумного будинку. В основному елементі управління використовується мікросхема ESP32-PICO, що інтегрована з Wi-Fi технологіями і має 4MB вбудовану флеш-пам'ять SPI. Містить вбудований USB-інтерфейс Type-C, що забезпечує швидке завантаження та виконання програм [35].

Порівняльна характеристика даних МК наведено в табл. 2.5.

Таблиця 2.5 – Порівняльна характеристика МК

Параметр	Arduino Uno	ESP8266	ESP32/ M5Stack Atom Lite
Bluetooth	Немає	Немає	Bluetooth: v4.2 BR/EDR та BLE
Типова частота	16 MHz	80 MHz	160 MHz
SRAM	2 КБ	160 Кб	520 Кб
GPIO	14	17	36
Hardware / Software PMW	14 / 20	Немає / 8	1 / 16
АЦП	6 (10 біт)	1 (10 біт)	18 (12 біт)
Напруга живлення	5 В	3.3 В	3.3 В
Напруга АЦП	5 В	1 В	3.3 В

Оскільки у кожного з МК різні параметри АЦП, в ході проведення тестувань необхідно на це звертати уваги задля правильності зчитування та уникнення виходу з ладу МК.

2.2.4 Параметри датчиків та модулів

В ході досліджень та тестувань було використано наступні компоненти:

- модуль датчика напруги змінного струму Zmpt101b 250V;
- датчик контролю наявності мережевої напруги 220В з оптопарою;
- датчик сили струму ZMCT116A;
- реле 3.3V-5V;
- перетворювач AC-DC 220V to 5V 3W.

Порівняльна характеристика компонентів та функцій системи віддаленого моніторингу електрозбереження наведено в табл. 2.6.

Таблиця 2.6 – Апаратні компоненти системи віддаленого моніторингу електрозбереження

Компонент	Функція
Модуль датчика напруги змінного струму Zmpt101b 250V	Зчитування показань напруги змінного струму в мережі. Використовується також для обчислення потужності споживача
Датчик контролю наявності мережевої напруги 220В з оптопарою	Перевірка наявності напруги в мережі
Датчик сили струму ZMCT116A	Зчитування показань сили струму, яку використовує споживач. Використовується також для обчислення потужності.
МК M5Stack ATOM	Забезпечення отримання даних з датчиків та відправка їх на віддалений сервер
Електромеханічне реле Songle SRU-05VDC-SL-C	Віддалений доступ до ввімкнення та вимкнення електроприладу
Модуль перетворювач HI-LINK HLK-PM01	Пониження напруги з 220 В до 5 В для можливості роботи системи від електромережі

Оскільки основою проєкта є робота з напругою 220 В, для живлення МК та датчиків необхідно отримати постійний струм напругою 5 В. Саме тому й було використано перетворювач HI-LINK HLK-PM01. Даний конвертер дозволяє отримати необхідну напругу з потужністю 3 Вт. Приклад даного модуля наведено на рис. 2.17 [36].



Рисунок 2.17 – Приклад модуля перетворювача [36]

Окрім власне контролю споживчої енергії пропонується й керувати електроприладом, шляхом відключення його від мережі в критичні моменти, необхідно також використати модуль електромеханічного реле на 1 канал з опторозв'язкою 5V - 250V 10A. Даний модуль дозволяє керувати МК стан увімкнення електроприладу. Приклад такого реле наведено на рис. 2.18 [37].



Рисунок 2.18 – Приклад модуля електромеханічного реле на 1 канал з опторозв'язкою 5V - 250V 10A [37]

Для контролю наявності струму доречно використати датчик контролю наявності мережевої напруги 220В. Оптопара в даному випадку буде розв'язкою, що зменшить можливість виникнення небезпечної ситуації. Робота датчика досить проста. Після підключення на платі датчика почне

горіти зелений світлодіод, позначений LED. За наявності напруги 220 В світлодіод LED горить, якщо така напруга відсутня – світлодіод LED не горітиме. Вихідний сигнал датчика при наявності напруги 220 В має значення близько 0,5 В постійного струму. Якщо напруги 220 немає – вихідний сигнал має значення логічного нуля. Даний модуль наведено на рис. 2.19 [38].



Рисунок 2.19 – Датчик контролю наявності мережевої напруги 220В з оптопарою [38]

Модуль датчика напруги змінного струму Zmpt101b 250V дозволяє отримати значення напруги у мережі за допомогою МК. Цей модуль має багатооборотний потенціометр, який можна використовувати для регулювання аналогового виходу. Модуль може вимірювати напругу змінного струму в межах 250 В, відповідний вихідний режим можна налаштувати. Вихідний сигнал має вигляд синусоїди, форма хвилі медіана. Даний модуль наведено на рис. 2.20 [39].

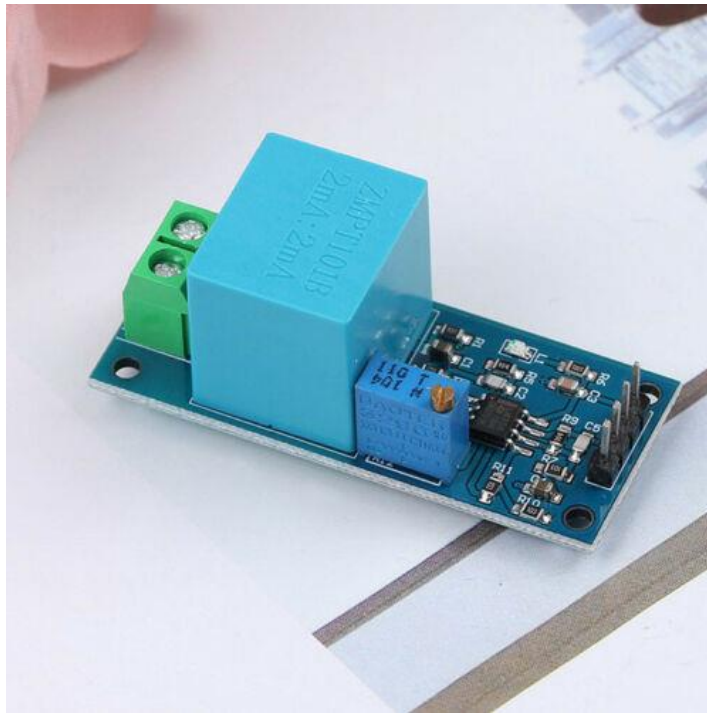


Рисунок 2.20 – Модуль датчика напруги змінного струму Zmpt101b 250V [39]

Найголовнішим датчиком є модуль датчика сили струму. В ході досліджень було виявлено 2 типи: інвазійного та неінвазійного типу. Інвазійний спосіб вимагає розрізання одного з провідників та підключення до нього, що дещо знижує безпеку. Неінвазійний спосіб працює на ефекті електромагнітного поля, що дозволяє зчитати показники сили струму не порушуючи ізоляцію провідника [40].

Датчик струму ACS712 30A GY-712 є гарним прикладом підключення інвазійним способом. Вимірювання струму до 30A, живлення 5V. Вихід підключається до аналогового входу МК і вимірюється надходження напруги з розрахунку 66 мВ на 1 А (тобто, наприклад, якщо на вході 132мВ, то протікає струм 2А) [40].

Залежно від включення може заміряти від 0 до 30А або від 0 до -30А. При вимірюванні змінного струму є нюанси - оскільки в цьому випадку в залежності від періоду синусоїди струм буде змінюватися при вимірах в різний час показання будуть відрізнятися, для коректного вимірювання потрібно робити виміри в піковій точці синусоїди. Даний модуль наведено на рис. 2.21 [40].



Рисунок 2.21 – Інвазивний датчик струму [40]

В якості неінвазивного способу використано датчик сили струму ZMCT103C. Має передаточне відношення обмотки котушки 2500:1. Напруга ізоляції становить 4500 В. Лінійний діапазон вимірювань становить до 70 А. Даний модуль наведено на рис. 2.22 [41].



SIGMA

Рисунок 2.22 – Датчик сили струму ZMCT103C [41]

Таким чином, в ході аналізу апаратних засобів підтримки процесу прототипування було досліджено особливості роботи МК Arduino Uno, ESP8266 та M5Stack Atom Lite на базі ESP32, а також було проаналізовано модуль датчика напруги змінного струму Zmpt101b 250V, датчик контролю наявності мережевої напруги 220В з оптопарою, модуль датчика сили струму ZMCT116A та електро механічне реле Songle SRU-05VDC-SL-C.

2.3 Висновки до розділу 2

Аналіз вимог до розроблюваного програмного та апаратного забезпечення з метою дослідження та розробки системи віддаленого моніторингу споживання електроенергії дозволив визначити місце розробки в архітектурі прототипу та обрати апаратні компоненти, середовища розробки, ОС та протокол взаємодії складових.

Таким чином, при розробці прототипу в якості МК виступає M5Stack ATOM. Він керує такими апаратними компонентами, як датчик сили струму, модуль електромеханічного реле, датчик напруги змінного струму. Для живлення самого МК використовується модуль перетворювача напруги.

Програмування МК виконується мовою C++ в середовищі програмування Arduino IDE.

Для віддаленого моніторингу даними електроспоживання використовується платформа OpenHAB з використанням протоколу передачі даних MQTT.

3 РОЗРОБКА ТА ДОСЛІДЖЕННЯ ПРОТОТИПУ СИСТЕМИ ВІДДАЛЕНОГО МОНІТОРИНГУ СПОЖИВАННЯ ЕЛЕКТРОЕНЕРГІЇ

3.1 Аналіз вимог до системи

В ході проведених досліджень було проаналізовано програмну та апаратну складові, що мають бути запровадженими у прототипі системи віддаленого моніторингу споживання електроенергії. Відповідно до виявлених особливостей, поставлено ряд вимог, що мають бути реалізовані в ході роботи. В результаті мають бути виконані наступні вимоги до проєкту:

- реєстрація наявності напруги в мережі;
- вимірювання поточної напруги змінного струму в мережі;
- вимірювання сили струму, яку споживає;
- на основі отриманих даних сили струму та напругу вимірювання значення потужності, яку споживає електроприлад;
- відправка обчислених значень на віддалений сервер для моніторингу користувачем та системою;
- перемикання стану електроприладу за допомогою реле під наявності такого повідомлення від серверу;
- задоволення вимог безпеки.

Для відправлення даних на сервер для віддаленого моніторингу системи контролю електроспоживання доречно використати МК з можливістю виходу до мережі Інтернет.

Також мікроконтролер має забезпечувати достатню точність отриманих показників, а тому має містити декілька вбудованих аналогових входів АЦП з розрядністю не менше 10 біт. Таким чином МК буде виконувати як функції зчитування та обробки даних, так і відправки до серверу.

Прототип повинен отримувати дані з мінімальною похибкою у короткі проміжки часу. Адже точність у масштабі цілої системи таких прототипів буде давати значущі дані, що можуть принести великі збитки в ситуації з великою похибкою даних. А невеликі проміжки часу нададуть системі можливість у

короткий час усувати проблеми, що також гарно позначитися на енергоефективності та безпеці.

На основі зчитаних даних сили струму споживача та величині напруги в мережі обчислювати потужність споживання енергії. Це надасть можливість використовувати прототип не тільки як лічильник спожитої енергії конкретним електроприладом, а й у якості контролеру стану мережі, що не дасть можливості виникнення аварійної ситуації в будинку, а також ситуації, що викличе спрацювання запобіжника, яка призведе до знеструмлення усього будинку.

Прототип має базуватися на схемі, що унеможливить імовірність аварійної ситуації з ним. Тобто сама схема має дотримуватися вимог безпеки, таких як гальванічна розв'язка чи надійна ізоляція компонентів. Самі компоненти мають бути з параметрами, що мають запас у порівнянні з тими параметрами, що пропонується вимірювати. Це унеможливить порушення безпеки системи в ситуації критичного стану, такому як стрибок напруги, порушення частоти в мережі чи мимовільне знеструмлення мережі.

Таким чином, в результаті роботи має бути забезпечено описані вимоги, що дозволить за допомогою прототипу дослідити системи віддаленого контролю та моніторингу споживання електроенергії.

3.2 Архітектура системи

Складові архітектури системи віддаленого моніторингу, наведені на рис. 3.1, відповідають узагальненій схемі трьохрівневої автоматизованої системи контролю та обліку енергоресурсів, наведеній в розділі 1.1.

Архітектура описує розташування та зв'язки між датчиками, мікроконтролером та віддаленим сервером. Така конструкція дозволяє забезпечити ефективну передачу та обробку даних, забезпечуючи користувачеві можливість моніторити споживання електроенергії в режимі реального часу.

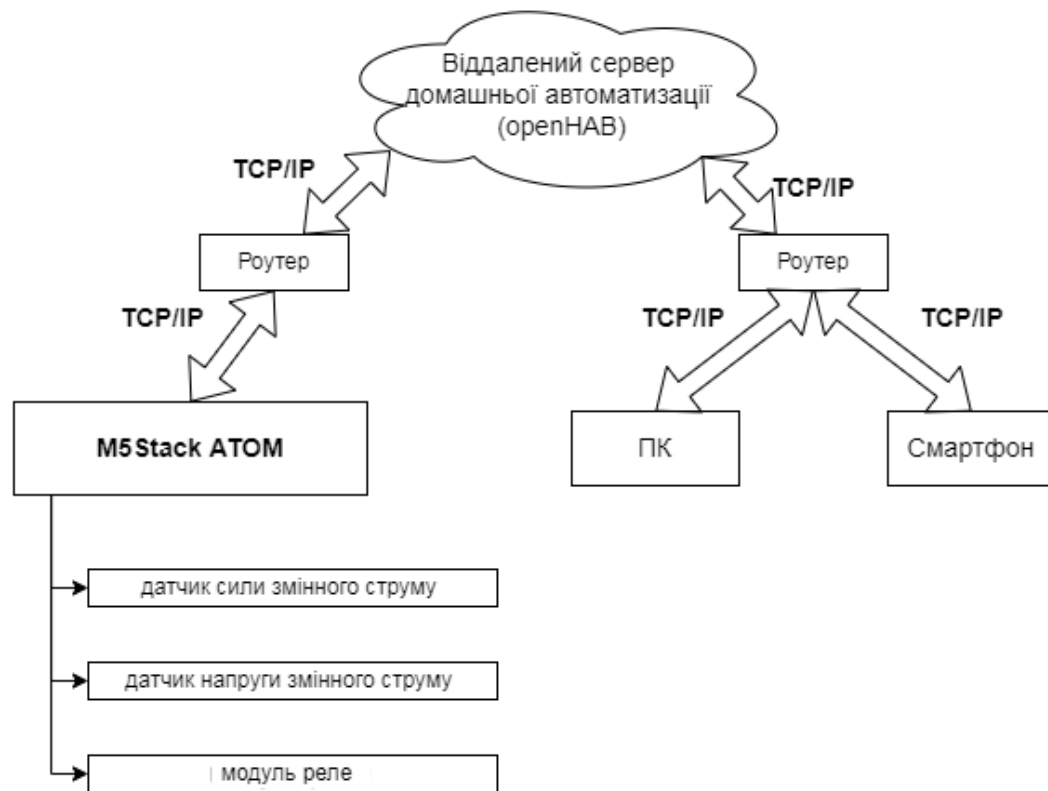


Рисунок 3.1 – Архітектура системи віддаленого моніторингу

Нижньому рівню відповідають модуль реле, датчик струму та датчик напруги. Саме вони зчитують такі дані як поточна сила струму споживання та напруга енергосистеми.

Ці дані від датчиків обчислюються за допомогою МК M5 Stack ATOM через дротовий зв'язок. Далі він на основі отриманих даних знаходить необхідні параметри системи за допомогою проведення фільтрації та обчислювання за формулами. Таким чином дана на даному етапі МК виступає в якості середнього рівня узагальненої схеми трьохрівневої автоматизованої системи контролю та обліку енергоресурсів.

Завдяки протоколу MQTT, що працює поверх протоколу TCP/IP, дані від МК передаються до роутера, який у свою чергу надсилає їх далі до віддаленого серверу домашньої автоматизації openHAB, що в свою чергу відповідає найвищому рівню схеми автоматизованої системи контролю та обліку енергоресурсів.

Далі вже знову за допомогою протоколу TCP/IP користувач може підключитися до системи віддаленого моніторингу за допомогою персонального комп'ютера (ПК) або телефону. Таким чином надається можливість спостерігати за енергоспоживанням.

Таким чином, архітектура складових системи віддаленого моніторингу складається з трьох рівнів, де першим є датчики, другим – мікроконтролер. А третій рівень – віддалений сервер домашньої автоматизації openHAB. Взаємодіючи між собою ці три рівні забезпечують цілісність роботи системи

3.3 Розробка та дослідження апаратного забезпечення системи

3.3.1 Загальна апаратна схема

В ході дослідження системи віддаленого моніторингу було використано ряд апаратних компонентів: датчик наявності напруги в мережі, модуль реле з оптопарою, датчик напруги та датчик струму, МК Arduino UNO, NodeMcu ESP8266 та M5Stack ATOM на базі ESP32. Приклад апаратної реалізації системи віддаленого моніторингу наведено на рис. 3.2.

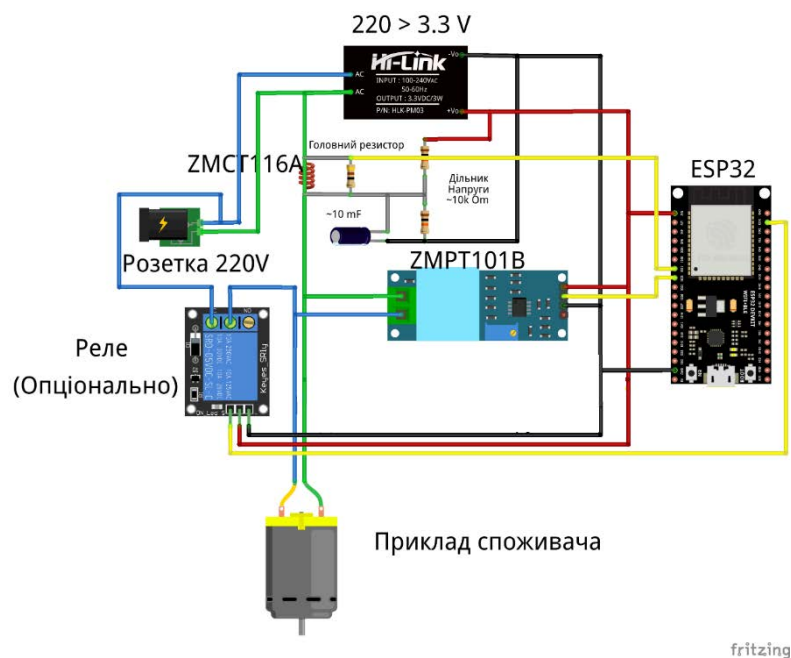


Рисунок 3.2 – Схема апаратної реалізації прототипу системи віддаленого моніторингу

3.3.2 Опис реалізації датчика контролю наявності мережевої напруги

Найпростішим модулем є датчик контролю наявності мережевої напруги 220В. Його під'єднання до мережі відбувається інвазійним способом, тобто через нього проходить струм мережі. Безпека забезпечується шляхом опторозв'язки.

Підключення до МК відбувається шляхом під'єднання до будь-якого з цифрових портів. Таким чином, при наявності струму в мережі на виході модуля буде логічна одиниця, а в протилежному випадку – сигнал низького рівня. Приклад експериментів над даним модулем наведено на рис. 3.3 – 3.4.

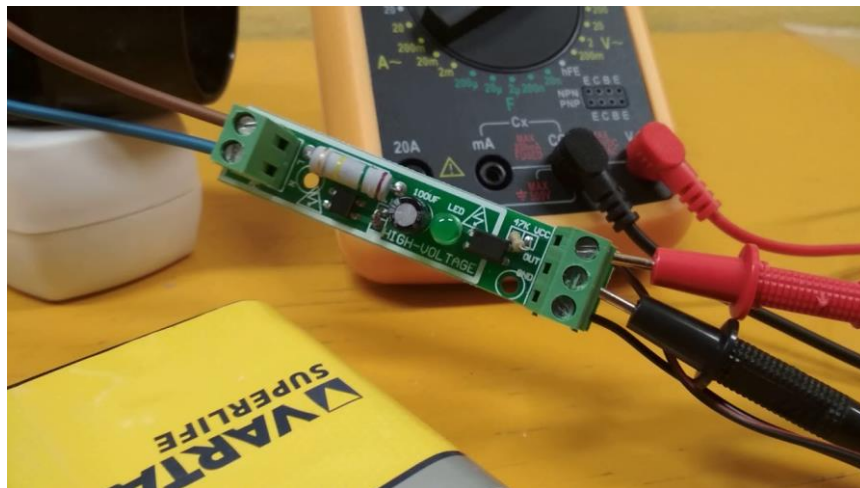


Рисунок 3.3 – Приклад експериментів з датчик контролю наявності мережевої напруги 220В

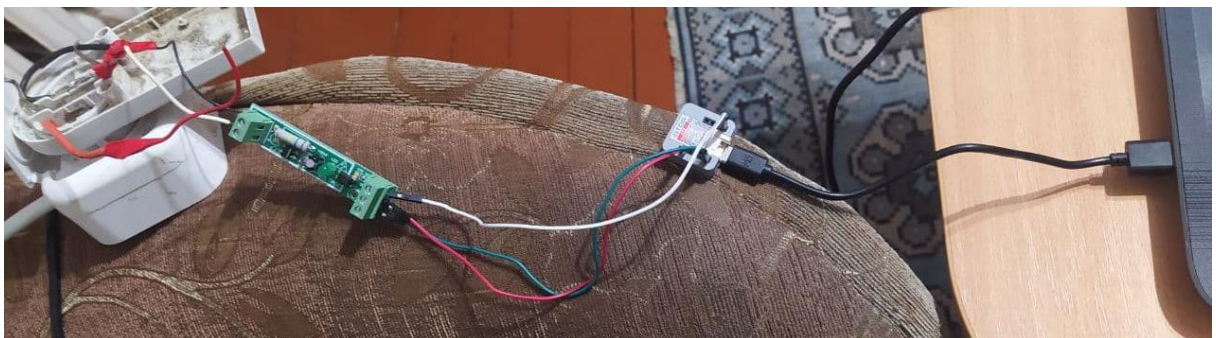


Рисунок 3.4 – Приклад експериментів з датчик контролю наявності мережевої напруги 220В

В результаті отримані від даного модуля дані для наочності на час тестів також виводяться у монітор порта. Приклад наведено на рис. 3.5.

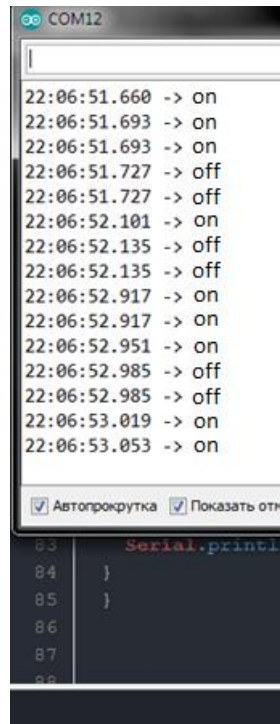


Рисунок 3.5 – Результати роботи датчика контролю наявності мережевої напруги 220В

3.3.3 Опис реалізації модуля реле з оптопарою

При підключенні реле спочатку необхідно обрати тип реле: високого рівня чи низького рівня. При підключенні реле можуть бути певні нюанси. Якщо блок реле має оптичну розв'язку, тобто з оптопарою, то, скоріш за все, нічого не доведеться переробляти, проте якщо блок реле без оптичної розв'язки, доведеться переробити роботу з GPIO, оскільки у перших реле включається низьким рівнем, а не високим, а по-друге високий рівень ESP8266 та ESP32 – це 3.3В, для 5-вольтового блоку реле цього не достатньо.

У випадку використання МК Arduino Uno проблем взагалі не має виникнути, адже номінальна напруга на пінах складає +5В.

Для підключення електромагнітного реле, що не має оптичної розв'язки та містить напругу керування відмінну від 3,3В до МК на базі ESP8266 та ESP32 доведеться налаштувати комутаційний вихід як OPENDRAIN, тобто дозволити піну працювати в режимі відкритого стоку. Також можна зробити невелику схему на базі транзистора NPN-типу, що також дозволить використовувати модуль реле. В такому випадку можна комутувати не тільки реле логічної напруги керування, а й більшої, наприклад 12В.

Приклад схеми підключення до МК ESP8266 та ESP32 на базі транзистора 2N2222 наведено на рис. 3.6.

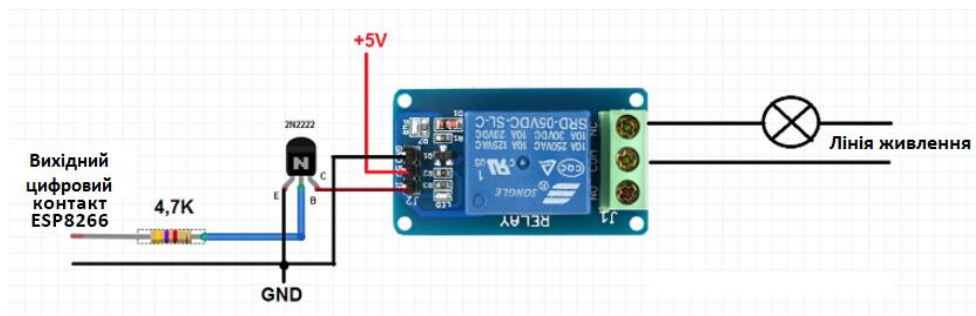


Рисунок 3.6 – Приклад схеми підключення модуля реле до МК ESP8266 та ESP32 на базі транзистора 2N2222

Приклад підключення модуля реле до МК ESP8266 та Arduino Uno наведено на рис. 3.7 та 3.8.

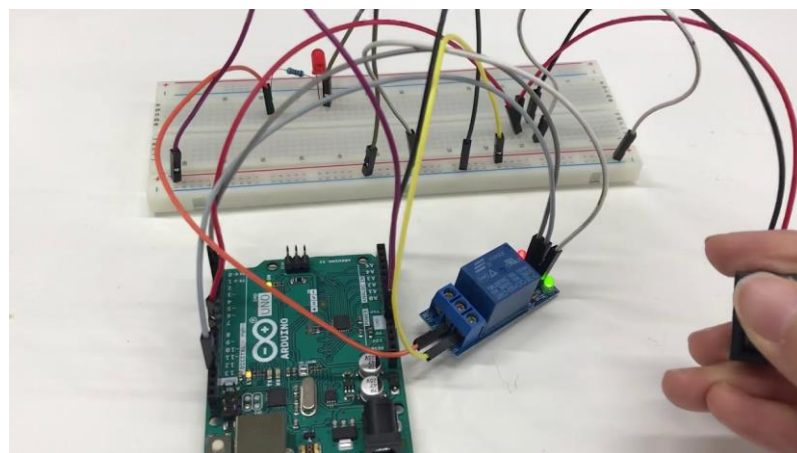


Рисунок 3.7 – Приклад підключення модуля реле до МК Arduino Uno

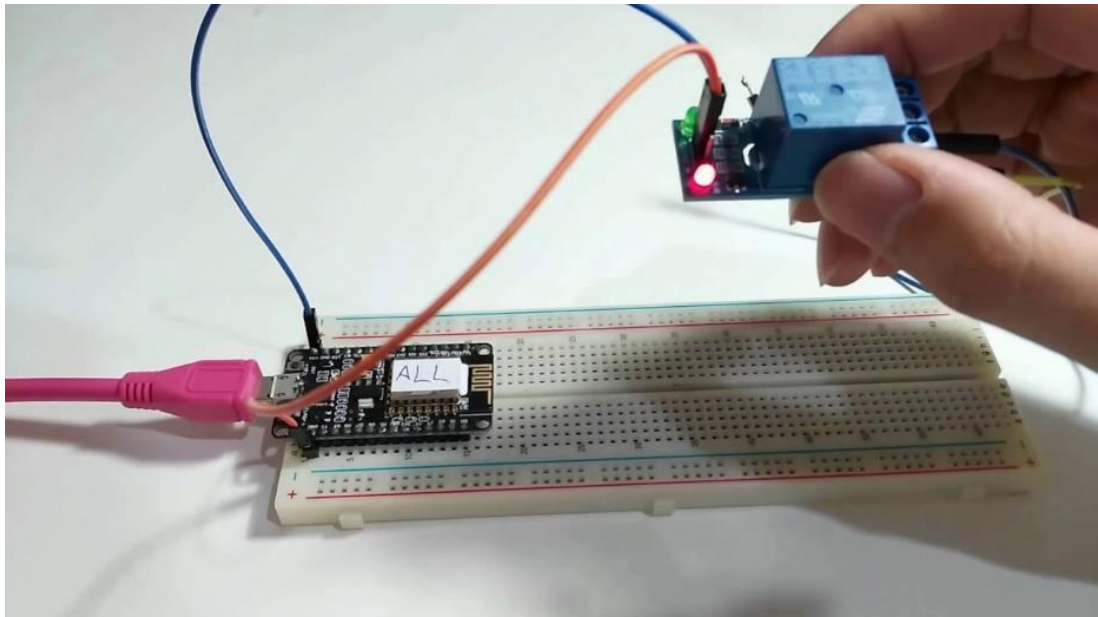


Рисунок 3.8 – Приклад підключення модуля реле до МК ESP8266

Таким чином, використовуючи модуль реле можна керувати станом ввімкнення та вимкнення електроприладу, підключеного до розумної розетки. Система домашньої автоматизації, використовуючи розумну розетку з модулем, зможе автоматично вимикати прилад у випадку критичної ситуації, наприклад при високому рівні навантаження на електросистему, що може призвести до пожежі. Також використання реле дозволить системі автоматично керувати станом електроприладу відповідно до заданого розкладу.

3.3.4 Опис реалізації модуля датчика напруги змінного струму Zmpt101b 250V

Також важливим є модуль датчика напруги змінного струму Zmpt101b 250V. Оскільки окрім стандартних 220 В в мережі можуть бути зміни в ту чи іншу сторону, показники електроспоживання без урахування даних з датчика можуть бути невірними. Це може призвести до небезпеки або до збитків. Тому й доречно використати даний модуль.

Модуль вимірювання напруги заснований на малопотужному трансформаторі. За його рахунок показники не спотворюються та повністю повторюють форму сигналу на виході сенсора. Датчик безпечний, оскільки має гальванічну розв'язку від високовольного входу. На виході датчика видається нормалізована синусоїда з додаванням постійної складової, тобто синусоїда міститься в рамках від нуля до напруги живлення датчика. Амплітуда цієї синусоїди підлаштовується за допомогою потенціометр на платі датчика.

Коли на виході немає навантаження, тобто нічого не підключено до входу, датчик має початкову напругу зміщення $V_{CC}/2$. Тобто якщо на вхід нічого не підключено і напруга живлення модуля 5 вольт, то на виході модуля буде 2,5 вольта, або ж половину від показників АЦП: 512 для Arduino Uno або 2048 для ESP32. Відповідні значення напруги до отриманих показників з АЦП для МК Arduino Uno і ESP32 наведено в табл. 3.1.

Таблиця 3.1 – Таблиця відповідності значень з МК Arduino Uno і ESP32 та результатів вимірювання з модуля датчика напруги змінного струму Zmpt101b 250V

Напруга		Модуль датчика напруги змінного струму Zmpt101b 250V В
Arduino Uno	0 В	-250
	2,5 В	0
	4,7 В	220
	5 В	250
ESP32	0 В	-250
	1,65 В	0
	3,102 В	220
	3,3 В	250

Графік відповідності значень з МК Arduino Uno та результатів вимірювання з модуля датчика напруги змінного струму Zmpt101b 250V наведено на рис. 3.9.

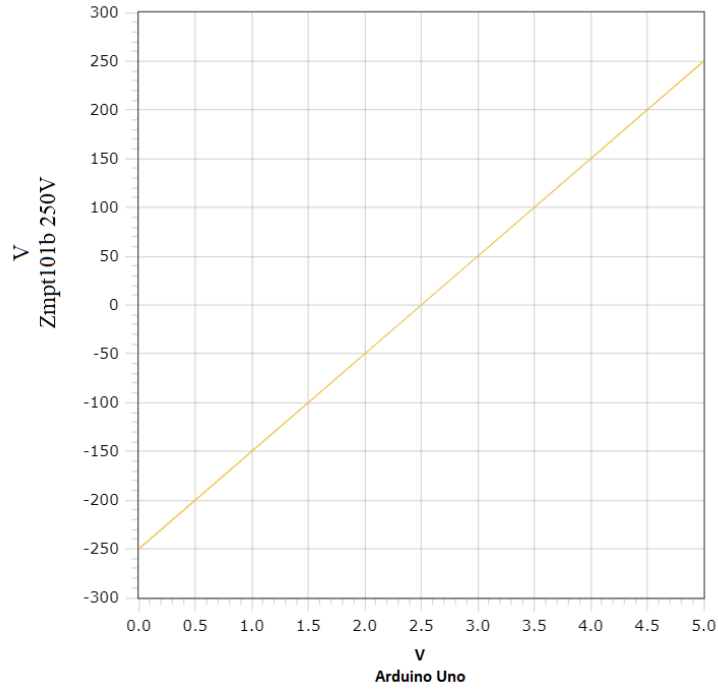


Рисунок 3.9 – Графік відповідності значень з МК Arduino Uno та результатів вимірювання з модуля датчика напруги змінного струму Zmpt101b 250V

Графік відповідності значень з МК ESP32 та результатів вимірювання з модуля датчика напруги змінного струму Zmpt101b 250V наведено на рис. 3.10.

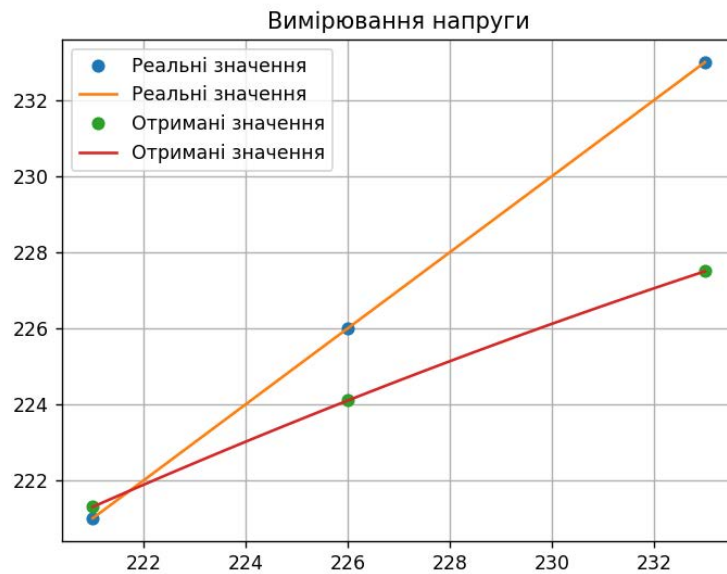


Рисунок 3.10 – Графік відповідності значень з МК ESP32 та результатів вимірювання з модуля датчика напруги змінного струму Zmpt101b 250V

Для знаходження значення напруги в мережі необхідно виконати обчислення. Спочатку знайдемо реальне амплітудне значення струму в мережі за формулою (3.1):

$$V_{out} = 250 \cdot \frac{V_s}{ADC_{max}}, \quad (3.1)$$

де V_{out} – амплітудне значення струму;

V_s – отримане значення з АЦП;

ADC_{max} – максимальне значення АЦП МК (1024 для Arduino Uno та 4096 для ESP32).

Таким чином, обчислене МК значення напруги з датчика необхідно поділити на квадратний корінь числа 2. Тобто знайти середньоквадратичне значення. Формула має наступний вид (3.2):

$$V = \frac{(V_{max} - V_{min}) \cdot 2}{\sqrt{2}}, \quad (3.2)$$

де V – значенням напруги в мережі;

V_{min} – максимальне отримане амплітудне значення струму;

V_{max} – мінімальне отримане амплітудне значення струму.

Калібрувати отримані значення необхідно за допомогою спеціалізованого обладнання, яке дозволить за допомогою потенціометра на платі скоригувати помилку. Приклади вимірювань реальним приладом, підключення модуля до ESP32 та Arduino, а також результати в моніторі порта наведені на рис. 3.12 та 3.13.

Таким чином, якщо під'єднати до МК модуль, можна побачити на послідовному плотері синусоїду. Приклад отриманої синусоїди з підключеного модуля до МК наведено на рис. 3.11.

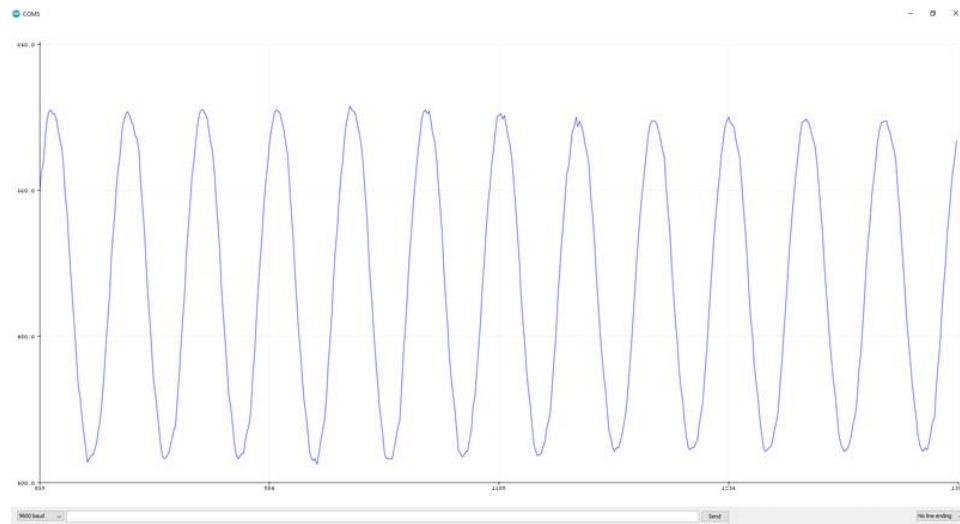


Рисунок 3.11 – Результати синусоїди на послідовному плотері модуля вимірювання напруги, підключеного до МК Arduino

Цей модуль вимірює розмах напруги. МК спочатку знаходить максимальне вимірне значення, тобто пікову напругу. Для визначення дійсного значення напруги необхідно обрахувати отримані дані, адже 220 В є значенням напруги, яка була б у системи з постійним струмом при тих же значеннях потужності. Реальне амплітудне значення струму в мережі становить приблизно 310 вольт.

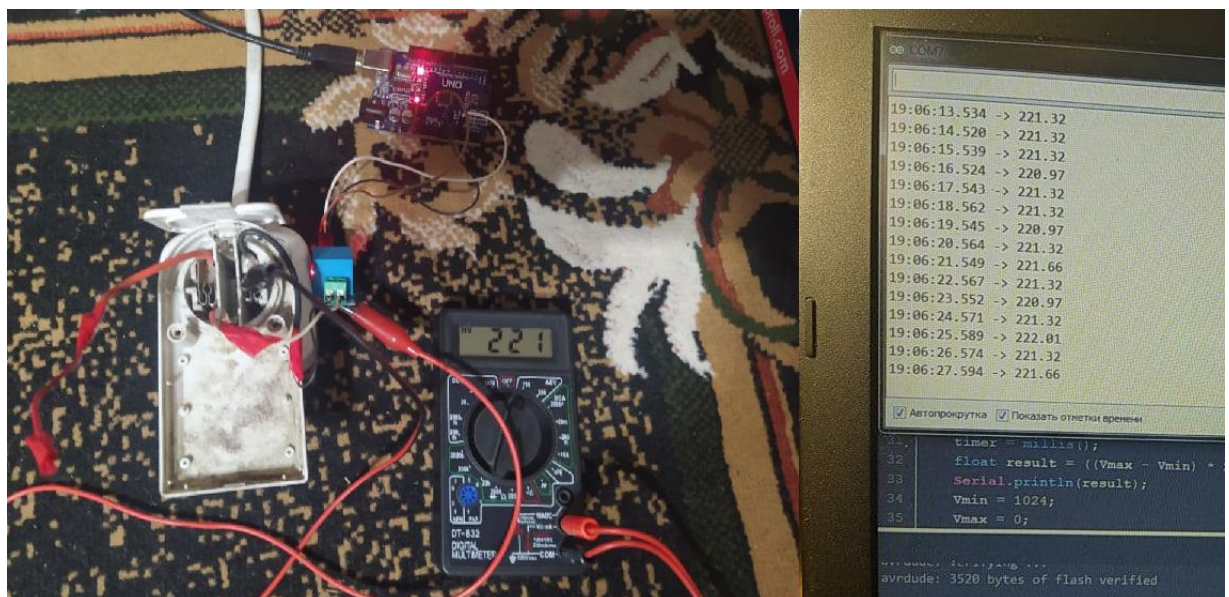


Рисунок 3.12 – Експерименти вимірювань модуля вимірювання напруги, підключеного до МК Arduino

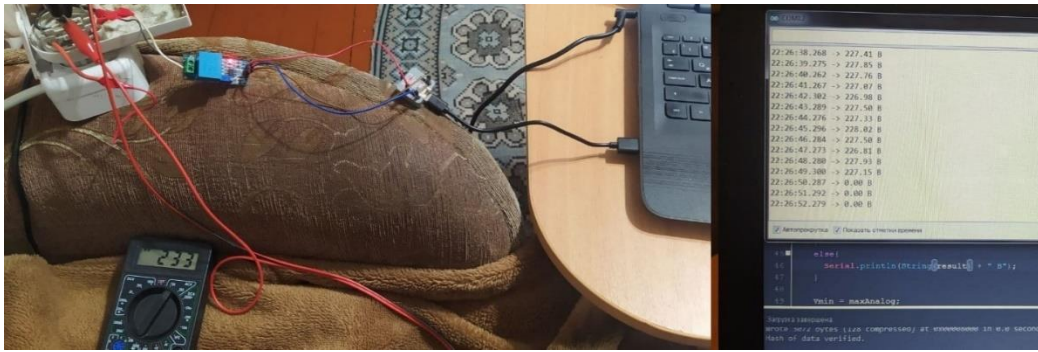


Рисунок 3.13 – Експерименти вимірювань модуля вимірювання напруги, підключеного до МК ESP32

Порівняння реальних і експериментальних значень напруги, виміряних в різні часові проміжки і на різних ділянках електромережі наведено в табл. 3.2.

Таблиця 3.2 – Таблиця реальних і експериментальних значень напруги, виміряних в різні часові проміжки і на різних ділянках електромережі

Реальне значення, В	Середнє значення, В	Похибка, %
221	221,3	0,136
226	224,1	0,841
233	227,5	2,36

Можна виділити відповідність точності похибки вимірювання значення напруги на одиницю АЦП для МК Arduino Uno і ESP32, що наведено в табл. 3.3.

Таблиця 3.3 – Таблиця відповідності точності похибки вимірювання значення напруги на одиницю АЦП для МК Arduino Uno і ESP32

	Точність	
МК	Вольт на одиницю АЦП, В	Відсоток, %
Arduino Uno	0,488	0,2
ESP32	0,122	0,05

3.3.5 Опис реалізації модуля датчика сили струму ZMCT103C

Найважливішим модулем є трансформатор струму ZMCT103C, адже саме на його основі отримується значення сили струму, яку споживає електроприлад, яка разом із напругою і становлять потужність. Саме тому даний датчик є основним у прототипі моніторингу електроенергії.

Модуль під'єднується неінвазійним способом, а тому це значно покращує безпеку.

Щоб підключити датчик до МК, вихідний сигнал модуля трансформатора струму повинен бути оброблений таким чином, щоб він відповідав вхідним вимогам аналогових входів конкретного МК, тобто позитивному напрузі між 0 В і опорною напругою АЦП. Таким чином досить важливо враховувати, який конкретно МК використовується, адже при неправильному підключенні в кращому випадку дані будуть невірними, а в гіршому – МК може вийти з ладу. Приклад схеми підключення для МК Arduino наведено на рис. 3.14 [42].

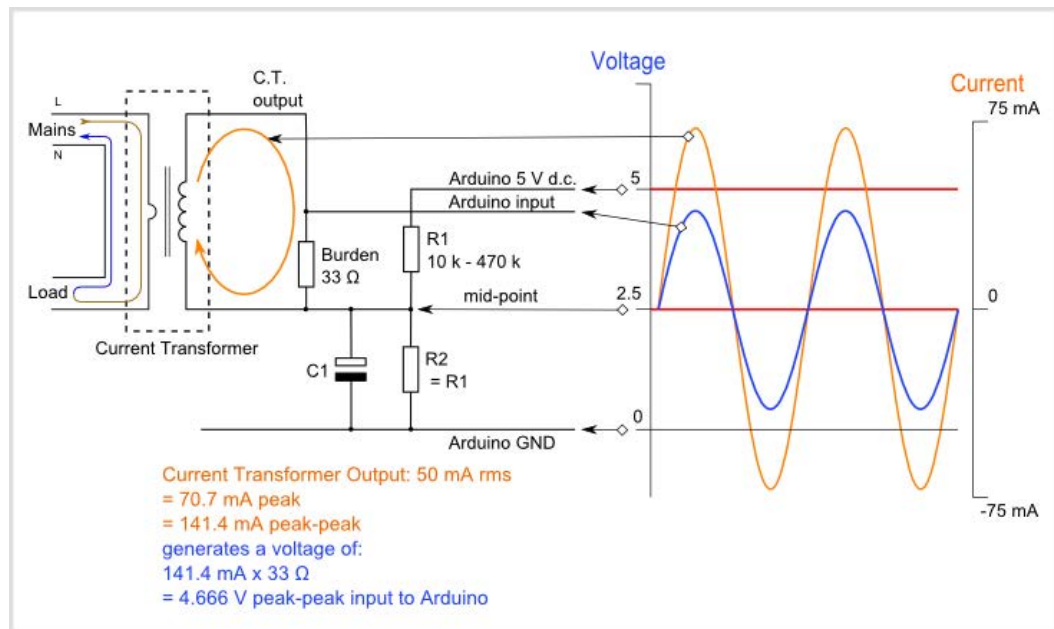


Рисунок 3.14 – Схема підключення модуля трансформатора струму до МК Arduino [42]

Правильних параметрів системи можна досягти за допомогою наступної схеми, що складається з двох основних частин:

- датчик трансформатору струму та навантажувальний резистор;
- дільник напруги зміщення R1 та R2.

Таким чином, для підключення до МК трансформатор струму ZMCT103C сигнал струму необхідно перетворити на сигнал напруги за допомогою навантажувального резистора. ZMCT103C має діапазон струму від 0 до 70 А. Тому 70 А – це максимальний струм.

Для знаходження необхідного номіналу резистора необхідно виконати обчислення. Спочатку знайдемо первинний початковий струм. Формула має наступний вид (3.3):

$$T_{пп} = \sqrt{2} \cdot T_c, \quad (3.3)$$

де $T_{пп}$ – первинний піковий струм;

T_c – максимальний струм (в даному випадку 70А).

Далі необхідно знайти піковий струм у вторинній обмотці. Формула має наступний вид (3.4):

$$T_{вп} = \frac{T_{пп}}{n}, \quad (3.4)$$

де $T_{вп}$ – вторинний піковий струм;

$T_{пп}$ – первинний піковий струм;

n – кількість витків обмотки (в даному випадку 2500).

Далі знаходимо ідеальне значення опору резистора. Формула має наступний вид (3.5):

$$R = \frac{A_{REF}}{2 \cdot T_{вп}} \quad (3.5)$$

де $T_{вп}$ – вторинний піковий струм;

R – ідеальне значення опору резистора;

$AREF$ – опорна напруга МК (наприклад 5 у Arduino та 3.3 у ESP32).

Таким чином доречно обрати відповідні значення резистору для МК Arduino Uno і ESP32, що наведено в табл. 3.4.

Таблиця 3.4 – Таблиця відповідності значення резистору для МК Arduino Uno і ESP32

Значення	Arduino Uno	ESP32
Резистор	63 Ом	41.7 Ом

Відповідно до рис. 3.4 схема прибирає від’ємну складову вихідного сигналу, піднімаючи значення напруги 0 до половини опорної напруги. Відповідно мінімальне амплітудне значення синусоїди буде становити 0А, а, відповідно, значення, що рівне $AREF - 70A$.

Значення половини $AREF$ отримуємо за допомогою звичайного дільника напруги, що виконується на двох резисторах однакового номіналу, бажано в діапазоні 10 кОм – 470 кОм.

Схема базового підключення такого датчика наведена на рис. 3.15 [42].

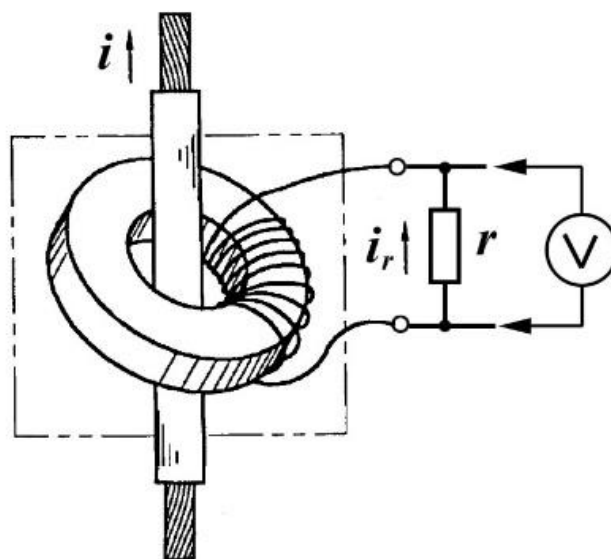


Рисунок 3.15 – Схема базового підключення датчика струму [42]

Загалом, схема досить проста, адже перетворення значення сили струму в напругу відбувається досить простим способом – за допомогою закону Ома.

Таким чином можна отримати значення синусоїди сили струму. Ці дані необхідно обчислити за допомогою формул. Спершу необхідно знайти значення напруги на виході датчика. Формула має наступний вид (3.6):

$$V_{pp} = \frac{V_m \cdot AREF}{ADCn}, \quad (3.6)$$

де V_{pp} - значення напруги на виході датчика;

V_m – максимальне отримане значення;

$ADCn$ – розрядність АЦП МК (наприклад 1024 у Arduino та 4096 у ESP32);

$AREF$ – опорна напруга МК.

Далі отримане значення напруги необхідно трансформувати у значення сили струму. Для цього необхідно за законом Ома знайти значення струму, вирахувати звідти пікову складову та врахувати кількість витків котушки. Формула має наступний вид (3.7):

$$I = \frac{V_{pp} \cdot n}{\sqrt{2} \cdot R}, \quad (3.7)$$

де V_{pp} - значення напруги на виході датчика;

R - ідеальне значення опору резистора;

n – кількість витків обмотки;

I – розраховане значення сили струму.

В ході експериментів для виявлення даних с датчика використовувався осцилограф. Приклад роботи осцилограф наведено на рис. 3.16.



Рисунок 3.16 – Підключення осцилографа в ході тестувань до схеми з датчиком струму

Можна виділити відповідність точності похибки вимірювання значення сили струму на одиницю АЦП для МК Arduino Uno і ESP32, що наведено в табл. 3.5.

Таблиця 3.5 – Таблиця відповідності точності похибки вимірювання значення напруги на одиницю АЦП для МК Arduino Uno і ESP32

МК	Точність	
	Ампер на одиницю АЦП, мА	Відсоток, %
Arduino Uno	0,137	0,2
ESP32	0,034	0,05

Графік відповідності значень з МК Arduino Uno і ESP32 та результатів вимірювання з модуля датчика сили струму ZMCT103C наведено на рис. 3.17.

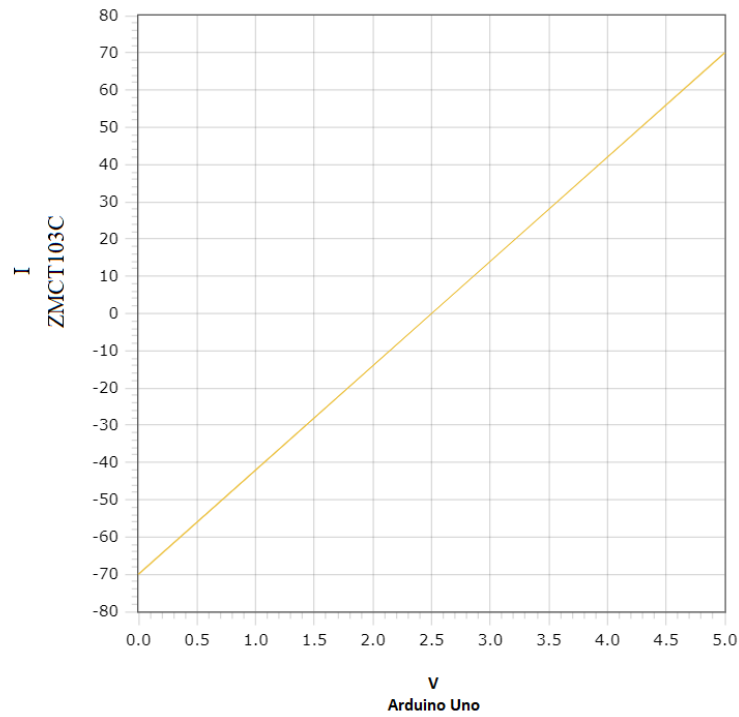


Рисунок 3.17 – Графік відповідності значень з МК Arduino Uno та результатів вимірювання з модуля датчика сили струму ZMCT103C



Рисунок 3.18 – Графік відповідності значень з МК ESP32 та результатів вимірювання з модуля датчика сили струму ZMCT103C

Порівняння реальних і експериментальних значень сили струму, виміряних в різні часові проміжки і на різних ділянках електромережі наведено в табл. 3.6.

Таблиця 3.6 – Таблиця реальних і експериментальних значень сили струму, виміряних в різні часові проміжки і на різних ділянках електромережі

Реальне значення, А	Середнє значення, А	Похибка, %
4.5	4.78	6,2
8.2	7.84	4.34
9.1	8.81	3.187

В результаті досліджень та у відповідності до формул 3.1 – 3.3 та рис. 3.14 було побудовано макет, який зображено на рис. 3.19 та 3.20.

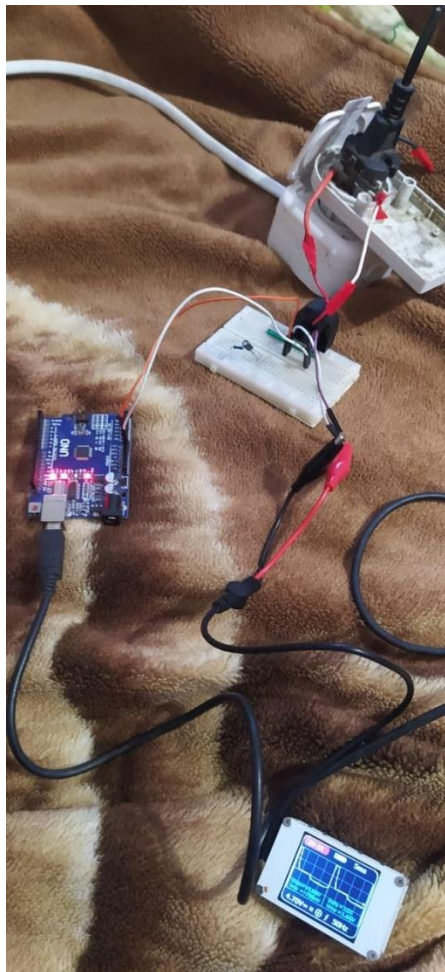


Рисунок 3.20 – Макет підключення датчика струму до МК Arduino

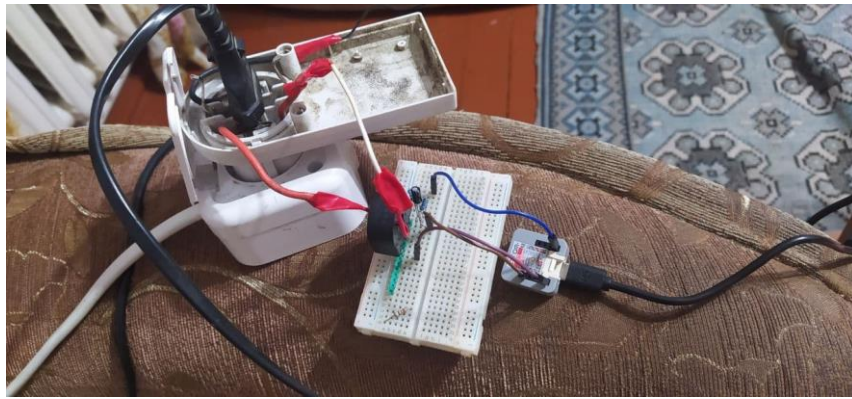


Рисунок 3.19 – Макет підключення датчика струму до МК ESP32

3.3.6 Збирання складових проєкту у прототип

Після налаштувань кожного з модулів окремо необхідно зібрати окремі компоненти до купи. Для цього необхідно розпаралелити виконання коду. Після збирання модулів відбулося тестування на прикладі електроприладу. Зовнішній вигляд прототипу наведено на рис. 3.21.



Рисунок 3.21 – Зовнішній вигляд прототипу після об'єднання модулів

Таким чином стала можливість отримувати повноцінний набір даних, де, окрім сили струму та напруги, можна спостерігати за потужністю споживання електроенергії електроприладом. Приклад отриманих результатів наведено на рис. 3.22.

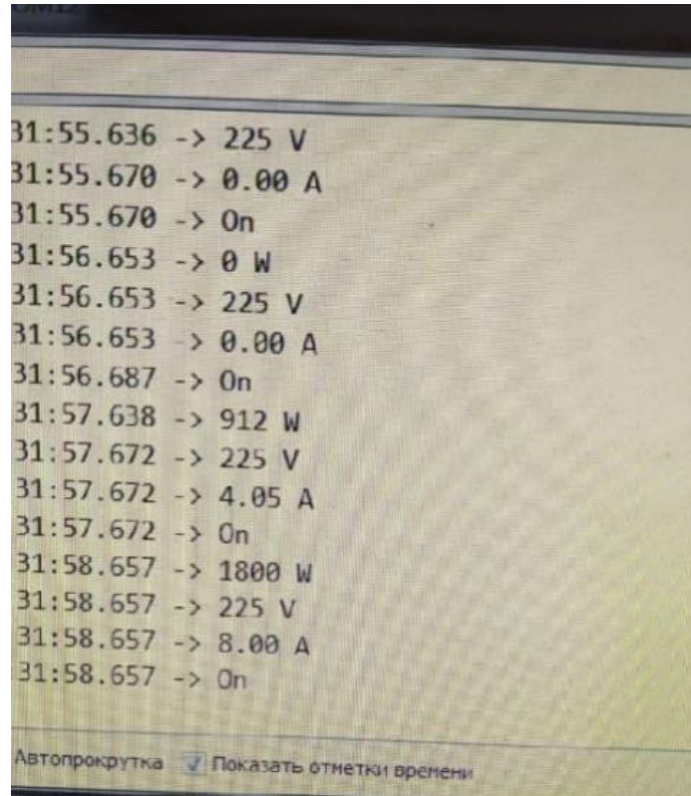


Рисунок 3.22 – Приклад отримання результатів прототипу

В результаті вимірювання значення сили струму та напруги можна виділити відповідність точності похибки вимірювання потужності на одиницю АЦП для МК Arduino Uno і ESP32, що наведено в табл. 3.7.

Таблиця 3.7 – Таблиця відповідності точності похибки вимірювання значення потужності на одиницю АЦП для МК Arduino Uno і ESP32

МК	Ватт на одиницю АЦП, Вт
Arduino Uno	0,067
ESP32	0,004

В ході досліджень було виявлено, що реальні значення потужності дещо відрізняються від реальних значень. Це зумовлено якістю конструкції, особливостями використаних радіодеталей та роздільної здатності МК. В результаті було побудовано таблицю відповідності реальної потужності до виміряної прототипом з використанням МК Arduino Uno і ESP32. Результати вимірювання потужності наведено на рис. 3.23.

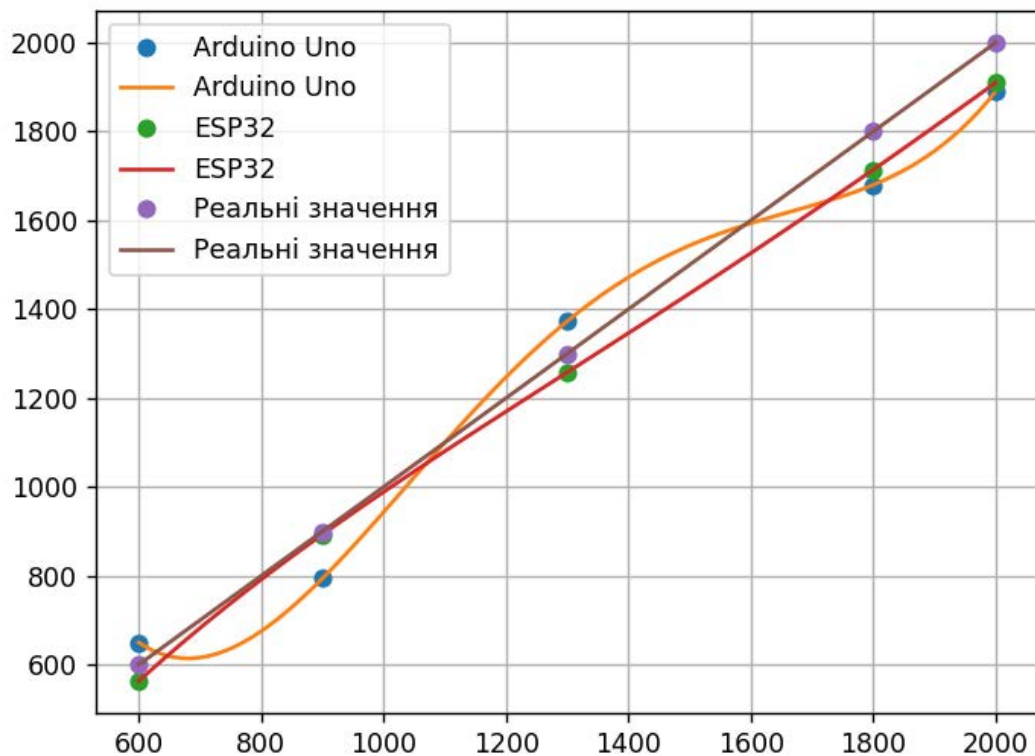


Рисунок 3.23 – Графік результатів вимірювання потужності з використанням МК Arduino Uno і ESP32

Як видно з рис. 3.23, реальні значення дещо відрізняються від отриманих. Таким чином можна зробити висновок, що різниця невелика, а тому в якості домашньої системи розумного лічильника даний проєкт можна використовувати. Проте для покращення результатів доречніше використати датчик струму з меншим запасом, наприклад на 20 ампер, що допустимо при використанні в будинку, де максимально допустима силу струму в 16 ампер.

Також можна покращити точність схеми за допомогою використання кращого датчика напруги, а також більш якісної схеми ділника напруги та використання апаратної фільтрації даних. Отримані показники наведено в табл. 3.8.

Таблиця 3.8 – Таблиця результатів вимірювання потужності з використанням МК Arduino Uno і ESP32

Реальне значення, Вт	Arduino Uno		ESP32	
	Значення, Вт	Похибка, %	Значення, Вт	Похибка, %
600	650	8,333	563	6,167
900	794	11,778	893	0,778
1300	1373	5,615	1258	3,231
1800	1680	6,667	1714	4,778
2000	1890	5,5	1910	4,5

Результат моніторингу на сервері отриманих даних наведено на рис. 3.24 та 3.25.

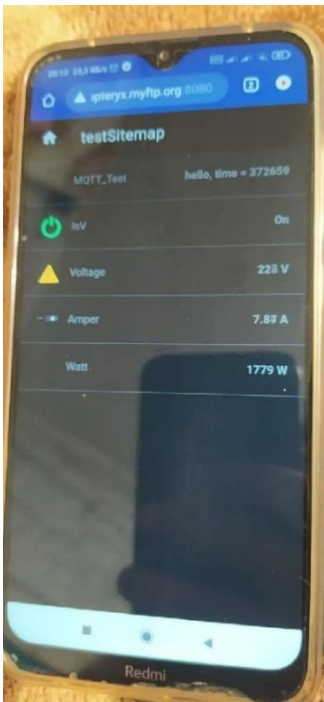


Рисунок 3.24 – Приклад моніторингу результатів роботи прототипу на сервері OpenHAB

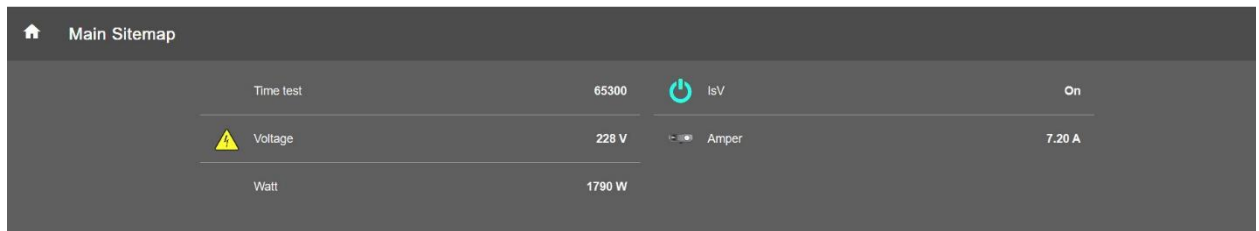


Рисунок 3.25 – Приклад моніторингу результатів роботи прототипу на сервері OpenHAB

Для віддаленого моніторингу необхідно отримані дані відправляти на сервер. Оскільки для моніторингу було обрано систему домашньої автоматизації OpenHAB, прототип має комунікувати з ним за допомогою MQTT.

Таким чином, в ході розробки та дослідження апаратного забезпечення системи віддаленого контролю та моніторингу споживання електроенергії було наведено загальну апаратну схему, описано та реалізовано роботу датчика контролю наявності мережевої напруги, модуля датчика напруги змінного струму Zmpt101b 250V, модуля датчика сили струму ZMCT103C, а також роботу модуля реле. В процесі розробки було зібрано тестовий прототип, протестовано різні сценарії використання та досліджено роботу даної системи в реальних умовах.

Крім того, під час тестування прототипу було виявлено, що додаткові вимірювання та аналіз точності можуть бути виконані для виправлення невеликих відхилень між реальними та отриманими значеннями.

Також, з огляду на результати тестування, можливе подальше дослідження та оптимізація системи з метою забезпечення оптимальної працездатності в різних умовах експлуатації.

Використання кращого датчика напруги та оптимізація схеми дільника напруги можуть сприяти підвищенню точності вимірювань, а вдосконалення апаратної фільтрації даних може допомогти усунути артефакти та забезпечити більш надійну інтерпретацію інформації.

3.4 Розробка ПЗ для системи віддаленого моніторингу

3.4.1 Алгоритм функціонування системи

Алгоритм функціонування системи наведено на рис. 3.26.

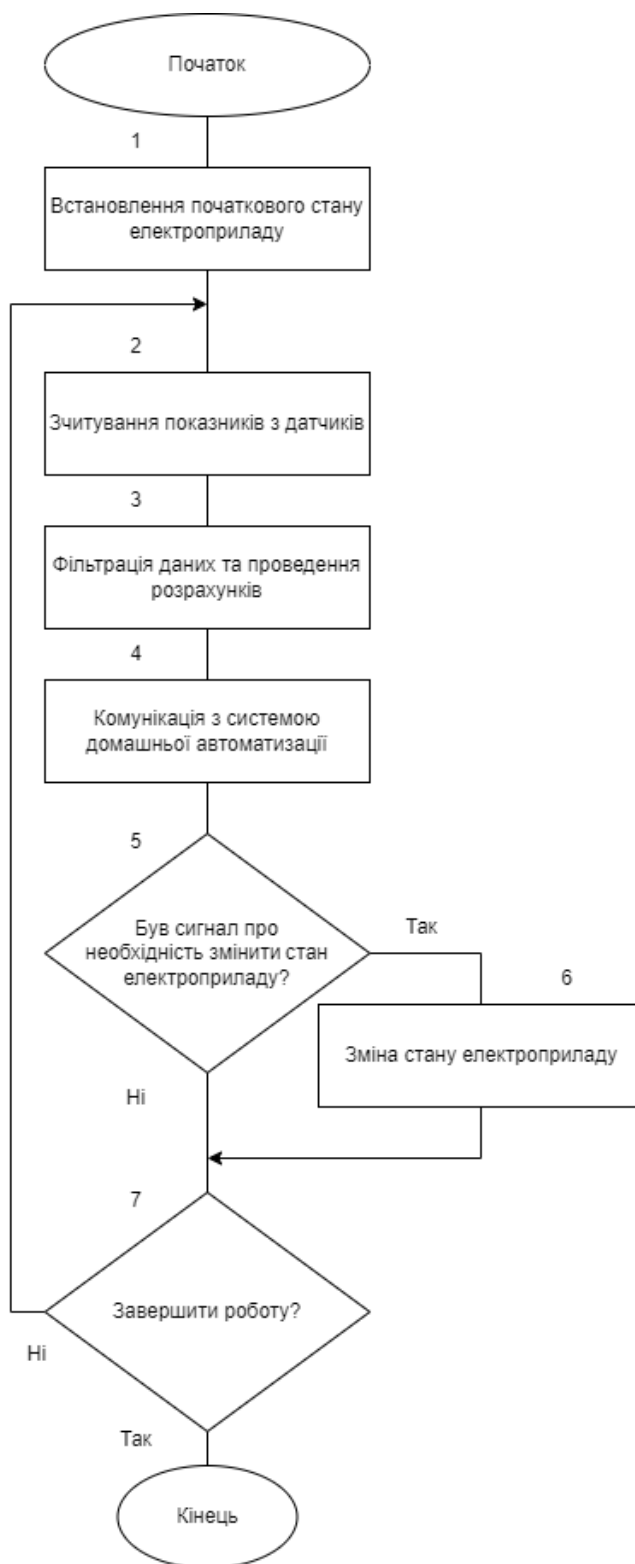


Рисунок 3.26 – Алгоритм функціонування системи віддаленого моніторингу

Відповідно до рис. 3.26 алгоритм функціонування системи віддаленого моніторингу складається з наступних блоків.

Блок 1. Встановлення початкового стану електроприладу. При запуску системи проводиться початкова ініціалізація всіх змінних, налаштування роботи передачі даних за допомогою WI-FI, а також стан електроприладу переводиться в зазначений при ініціалізації;

Блок 2. Зчитування показників з датчиків. Програма виконує зчитування даних з усіх датчиків та модулів для подальшої обробки;

Блок 3. Фільтрація даних та проведення розрахунків. Отримані на попередньому кроці дані фільтруються алгоритмом бігучого середнього арифметичного у відповідних функціях. Після проведення фільтрації отримані значення обчислюються за допомогою відповідних формул;

Блок 4. Комунікація з системою домашньої автоматизації. Після зчитування даних з датчиків, проведення фільтрацій та обчислень, отримані значення відправляються до сервера системи домашньої автоматизації. Також на цьому кроці програма від сервера отримує значення, яким має бути стан електроприладу;

Блок 5. Був сигнал про необхідність змінити стан електроприладу? На даному кроці аналізується отримане на попередньому кроці значення необхідного стану електроприладу. Якщо значення було змінено – програма переходить до кроку 6. В протилежному випадку – до кроку 7;

Блок 6. Зміна стану електроприладу. Якщо на попередньому кроці значення змінної стану електроприладі змінилося – програма перемикає стан електроприладу на протилежний;

Блок 7. Завершити роботу? При необхідності продовжити роботу програма повернеться на крок 2, в протилежному випадку завершить виконання.

Таким чином, алгоритм функціонування системи віддаленого моніторингу містить ряд кроків, такі як ініціалізація, зчитування показників, фільтрація та обчислення, комунікація з системою домашньої автоматизації та

зміна стану електроприладу розглядаються як етапи, що забезпечують надійну та ефективну роботу системи. Даний алгоритм створює можливість динамічного реагування на зміни та автоматизованого керування, що робить його важливим інструментом для покращення управління електроживленням та використанням електроприладів.

3.4.2 Функції та змінні програми

У таблиці 3.9 описані функції та змінні програми. Текст програми наведено в Додатку А.

Таблиця 3.9 – функції та змінні програми.

Назва функції або змінної	Опис
1	2
void setup()	Функція, що викликається один раз при запуску системи і виконує початкову ініціалізацію системи
void loop()	Функція, що викликається постійно після виклику setup() і в якій відбувається робота програми
void changeRelayState()	Функція, яка відповідає за зміну стану реле
double getVoltageValue()	Функція, яка відповідає за отримання поточного значення напруги
double getAmpereValue()	Функція, яка відповідає за отримання поточного значення сили струму
float isAvailabilityVoltage()	Функція, яка відповідає за отримання значення наявності напруги
float runMiddleArifmOptim(float)	Функція, яка відповідає за фільтрацію значення сили струму алгоритмом бігучого середнього арифметичного
void sendMessageMQTT(float, float, bool)	Функція, яка відповідає за відправку обчислених значень до сервера

Продовження таблиці 3.9

1	2
<code>float runMiddleArifmOptimV(float)</code>	Функція, яка відповідає за фільтрацію значення напруги алгоритмом бігучого середнього арифметичного
<code>void setup_wifi()</code>	Функція, яка відповідає за запуск Wi-Fi
<code>void callback(String, byte *, unsigned int)</code>	Функція, яка відповідає за отримання повідомлень від сервера
<code>void reconnect()</code>	Функція, яка відповідає за перезапуск підключення MQTT
<code>void serialPrint(float, float, bool)</code>	Функція, яка відповідає за виведення обчислених значень у послідовний порт
<code>#define AMPERE_SENSOR_PIN</code>	Константа, що зберігає значення піну підключення датчика сили струму
<code>#define RELAY_PIN</code>	Константа, що зберігає значення піну підключення модуля реле
<code>#define VOLTAGE_SENSOR_PIN</code>	Константа, що зберігає значення піну підключення датчика напруги
<code>#define VOLTAGE_AVAILABILITY_PIN</code>	Константа, що зберігає значення піну підключення датчика наявності напруги
<code>const int FILTER_ELEM_COUNT</code>	Константа, що зберігає значення кількості елементів вибірки для фільтрації
<code>const float MAX_ANALOG_READ</code>	Константа, що зберігає максимальне значення, яке може бути отримане з АЦП
<code>const float MAX_ANALOG_READ_V</code>	Константа, що зберігає максимальне значення напруги роботи МК
<code>const int COUNTS_OF_TURN</code>	Константа, що зберігає значення кількості витків котушки датчика струму
<code>const int RESISTANS</code>	Константа, що зберігає значення опору резистора датчика струму
<code>const int MAX_AMPESE_COIL</code>	Константа, що зберігає максимальне значення сили струму, яку може зчитати датчик струму
<code>uint32_t mainTimer</code>	Змінна, що зберігає поточне значення часу, яке пройшло з початку роботи програми

Продовження таблиці 3.9

1	2
int isTurnRelay	Змінна, що зберігає поточний стан реле
const char *WIFI_SSID_NAME	Константа, що зберігає значення назви точки доступу
const char *WIFI_PASSWORD	Константа, що зберігає значення паролю точки доступу
const char *MQTT_SERVER	Константа, що зберігає значення адреси MQTT сервера
const int MQTT_PORT	Константа, що зберігає значення порту сервера MQTT
const String TOPIC_NAME_SYNC_T	Константа, що зберігає значення назви топика MQTT сервера, яке відповідає за час синхронізації
const String TOPIC_NAME_IS_V	Константа, що зберігає значення назви топика MQTT сервера, яке відповідає за наявність мережевої напруги
const String TOPIC_NAME_A	Константа, що зберігає значення назви топика MQTT сервера, яке відповідає за значення силу струму
const String TOPIC_NAME_V	Константа, що зберігає значення назви топика MQTT сервера, яке відповідає за значення мережевої напруги
const String TOPIC_NAME_W	Константа, що зберігає значення назви топика MQTT сервера, яке відповідає за значення потужності
const String TOPIC_NAME_RELAY	Константа, що зберігає значення назви топика MQTT сервера, яке відповідає за значення поточного стану реле
WiFiClient espClient	Об'єкт класу WiFiClient, який використовується для взаємодії з WiFi
PubSubClient client(WiFiClient)	Об'єкт класу PubSubClient, який використовується для взаємодії з сервером MQTT

Продовження таблиці 3.9

1	2
<code>void Serial.begin(int)</code>	Функція, яка встановлює швидкість передачі даних через серійний порт
<code>template <typename T> T Serial.println(T)</code>	Функція, яка відповідає за виведення даних у серійний порт
<code>void pinMode(int, int)</code>	Функція, яка відповідає за для встановлення режиму роботи вводу/виводу для конкретного піна
<code>void digitalWrite(int, int)</code>	Функція, яка відповідає за використовується для установки цифрового сигналу на конкретний пін
<code>int analogRead(int)</code>	Функція, яка відповідає за зчитування аналогового сигналу з аналогового входу МК
<code>void delay(int)</code>	Функція, яка відповідає за затримку виконання програми на певний проміжок часу
<code>unsigned long millis()</code>	Функція, яка повертає кількість мілісекунд, що пройшли з моменту запуску МК
<code>double currentAmpereValue</code>	Змінна, що зберігає обчислене значення сили струму
<code>double currentVoltageValue</code>	Змінна, що зберігає обчислене значення напруги
<code>bool voltageAvailability</code>	Змінна, що зберігає значення наявності напруги
<code>int sensor_Value</code>	Змінна, що зберігає значення, отримане від цифрового піна
<code>float v_Out</code>	Змінна, що зберігає обчислене значення напруги, отримане від цифрового піна
<code>static uint32_t timerV</code>	Статична змінна, що відповідає за таймер роботи датчика напруги
<code>static float Vmin</code>	Статична змінна, що зберігає мінімальне значення, для обчислення значення напруги
<code>static float Vmax</code>	Статична змінна, що зберігає максимальне значення, для обчислення значення напруги
<code>static float result_I</code>	Статична змінна, що використовується для знаходження результуючого значення сили струму

Продовження таблиці 3.9

1	2
static uint32_t timerA1	Статична змінна таймеру для знаходження значення сили струму
static uint32_t timerA2	Статична змінна таймеру для знаходження значення сили струму
static int maxICur	Статична змінна, що використовується для знаходження усередненого значення сили струму
static int maxI	Статична змінна, що використовується для знаходження усередненого значення сили струму
static float result_V	Статична змінна, що зберігає результуюче значення після обчислень даних напруги
float res	Змінна, яка використовується при обчисленні значення сили струму
float nCurrThruResistorPP	Змінна, яка використовується при обчисленні значення сили струму
float nCurrThruResistorRMS	Змінна, яка використовується при обчисленні значення сили струму
int valuePinIsV	Змінна, що зберігає отримане значення від цифрового піна датчика наявності напруги
bool resultIsVoltage	Змінна, що зберігає значення наявності напруги
static int t	Статична змінна, що використовується для знаходження фільтруючого значення сили струму
static float vals[]	Статична змінна, що використовується для знаходження фільтруючого значення сили струму
static float average	Статична змінна, що використовується для знаходження фільтруючого значення сили струму
float result	Змінна, що зберігає результуюче значення сили струму після фільтрації
static int t2	Статична змінна, що використовується для знаходження фільтруючого значення напруги
static float vals2[]	Статична змінна, що використовується для знаходження фільтруючого значення напруги
float powerValue	Змінна, що зберігає значення потужності роботи

Кінець таблиці 3.9

1	2
static float average2	Статична змінна, що використовується для знаходження фільтруючого значення напруги
float result	Змінна, що зберігає результуюче напруги струму після фільтрації
String synchTimeMessage	Змінна, що зберігає текстове значення таймера синхронізації
String isV	Змінна, що зберігає текстове значення наявності напруги
String valueV_str	Змінна, що зберігає текстове значення напруги
String valueI_str	Змінна, що зберігає текстове значення сили струму
String valueW_str	Змінна, що зберігає текстове значення потужності
String messageTemp	Змінна, що зберігає текстове значення, отримане від сервера MQTT

Таким чином, в ході розробки ПЗ для системи віддаленого моніторингу було наведено алгоритм функціонування, який відображає роботу програми. Також було описано функції та змінні, які використовувалися в програмі.

3.5 Висновки до розділу 3

В ході проведення розробки та дослідженні прототипу системи віддаленого моніторингу споживання електроенергії було описано аналіз вимог та архітектуру системи, розроблено програмне та апаратне забезпечення. Під час тестувань було описано приклади роботи системи.

Також було описано роботу складових програмно-апаратного забезпечення, наведено алгоритм функціонування, а також описано функції та змінні, які використовувалися в програмі.

Алгоритм функціонування системи віддаленого моніторингу включає кроки ініціалізації, зчитування показників, фільтрації та обчислень, комунікації з системою домашньої автоматизації та зміни стану електроприладу. Цей алгоритм створює можливість динамічного реагування на зміни та автоматизованого керування, що робить його важливим інструментом для оптимізації управління електроживленням та використанням електроприладів.

Була представлена загальна схема, описано роботу датчиків та модулів, та зібрано тестовий прототип. Тестування дозволило виявити деякі відхилення між реальними та виміряними значеннями.

З огляду на результати тестування, можливі подальші дослідження та оптимізація системи для забезпечення оптимальної працездатності в різних умовах експлуатації. Використання кращого датчика напруги, оптимізація схеми дільника напруги та вдосконалення апаратної фільтрації можуть підвищити точність вимірювань та забезпечити більш надійну інтерпретацію інформації.

Для подальшого вдосконалення програмної частини системи віддаленого моніторингу можна розглядати кілька напрямків оптимізації та розвитку. По-перше, важливо провести глибший аналіз алгоритмів фільтрації та обчислень з метою їх оптимізації та підвищення ефективності. Це може включати в себе вдосконалення використовуваних математичних моделей та методів обробки даних.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування програмно-апаратного забезпечення

Проведена робота та розроблені програмні та апаратні компоненти виступають в якості прототипу системи віддаленого моніторингу.

Основне призначення – розробити систему віддаленого моніторингу споживання електроенергії для підвищення якості споживання електроенергії, покращення безпеки при використанні системи.

Розроблене програмно-апаратне забезпечення стосується системи віддаленого моніторингу. В результаті роботи налагоджено роботу прототипу у відповідності до поставлених вимог.

Для використання системи віддаленого моніторингу необхідно мати комп'ютер з швидкісним Інтернет-з'єднанням, що задовольняє наступні апаратно-технічні вимоги:

- ОС Microsoft Windows 7 або вище;
- процесор з частотою 0.1 ГГц або вище;
- RAM 512 МБ або вище;
- відеокарта з підтримкою роздільної здатності екрану 1280 x 720 пікселів та вище;
- швидкісне інтернет-з'єднання від 1 Мбіт/с або вище;
- клавіатура та миша.

4.2 Характеристики програми

Розробка програмних застосунків для плат проводилася високорівневою мовою програмування C++ з використанням інтегрованого середовища розробки для створення та завантаження програм на Arduino-сумісні плати. Розроблені програмні застосунки призначені для усіх Arduino-сумісних плат, зокрема для плати M5Stack Atom.

Розроблена функція зчитування наявності струму в мережі дозволяє перевірити, чи є взагалі в мережі струм, або чи під'єднано прототип до мережі. Це дозволить уникнути проблем зі зчитувань пустих даних.

Розроблена функція зчитування напруги змінного струму у мережі дає можливість оцінити стан мережі. Наприклад повідомити користувача про неправильні відхилення від норми. А також ці дані використовуються при розрахунку потужності.

Функція зчитування сили змінного струму направлена на оцінку потужності використання електроенергії споживачем. Разом з датчиком напруги вираховується потужність, що власне й допомагає користувачеві моніторити стан системи.

Прототип системи віддаленого моніторингу готовий до інтеграції в OpenHAB за інтерфейсом MQTT та надає можливість проводити наступні дослідження:

- вимірювати наявність напруги в мережі;
- вимірювати силу струму;
- вимірювати напругу мережі;
- вимірювати потужність використання електроенергії споживачем.

4.3 Звернення до програми

Для віддаленого моніторингу за станом електромережі необхідно мати доступ до мережі Інтернет. Необхідно перейти до сайту з вашою системою домашньої автоматизації openHAB або запустити відповідний додаток на смартфоні.

Відповідно до розташування користувачем показань даних на формі буде відображено отримані дані від системи.

4.4 Вхідні та вихідні дані

В якості вхідних даних виступають показники значень датчиків, що зчитуються МК, які потім передаються до OpenHAB.

Вихідними даними також є оброблені дані, що демонструють функціонування системи контролю електромережі.

Вхідні дані являються сирими показниками з датчиків, а тому оброблюються МК у відповідності до необхідних алгоритмів. Приклад отримання вхідних даних при налагодженні програмістом системи наведено на рис. 4.1 та 4.2.

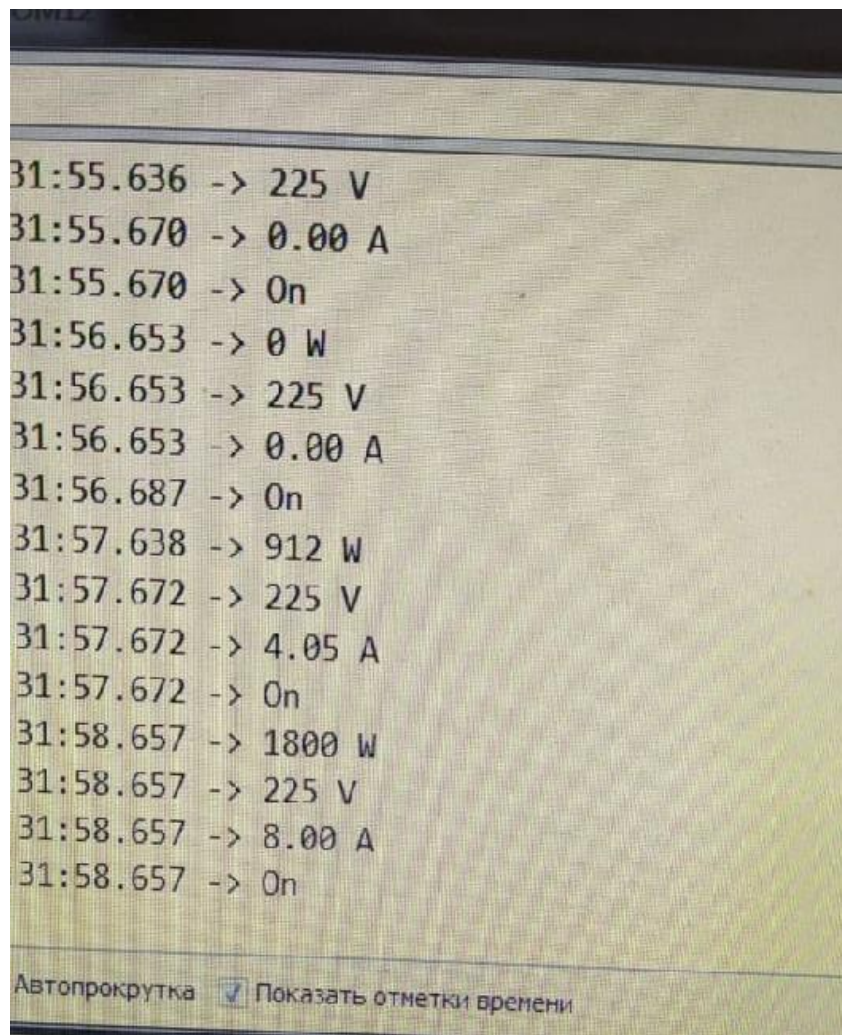


Рисунок 4.1 – Приклад налагодження системи та виведення проміжних даних у монітор порту

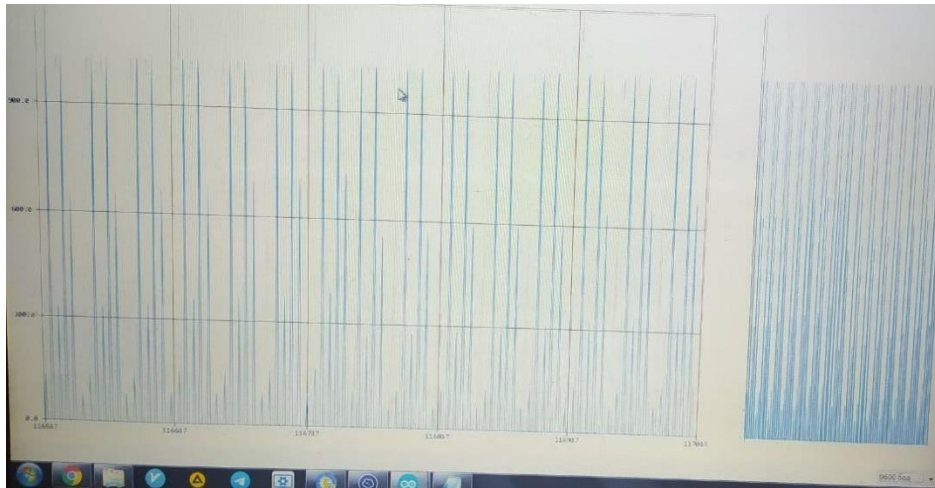


Рисунок 4.2 – Приклад отримання сирих даних від датчика

Вихідні дані можуть відображатися на формі платформи OpenHAB. Приклад отримання вихідних даних наведено на рис. 4.3 та 4.4.



Рисунок 4.3 – Приклад відображення вихідних даних на смартфоні

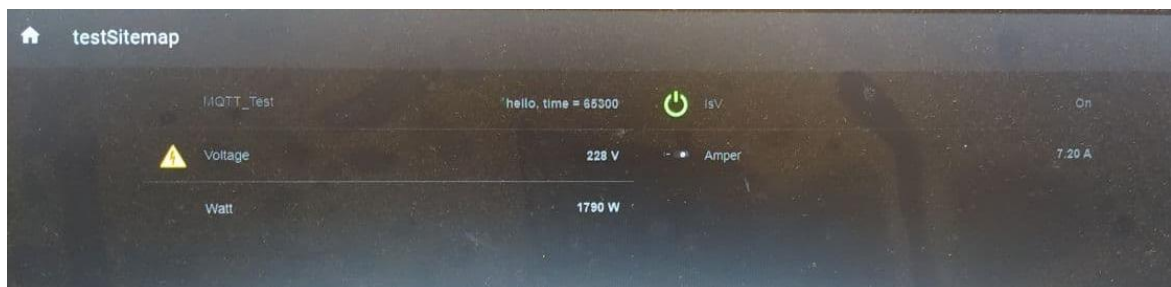
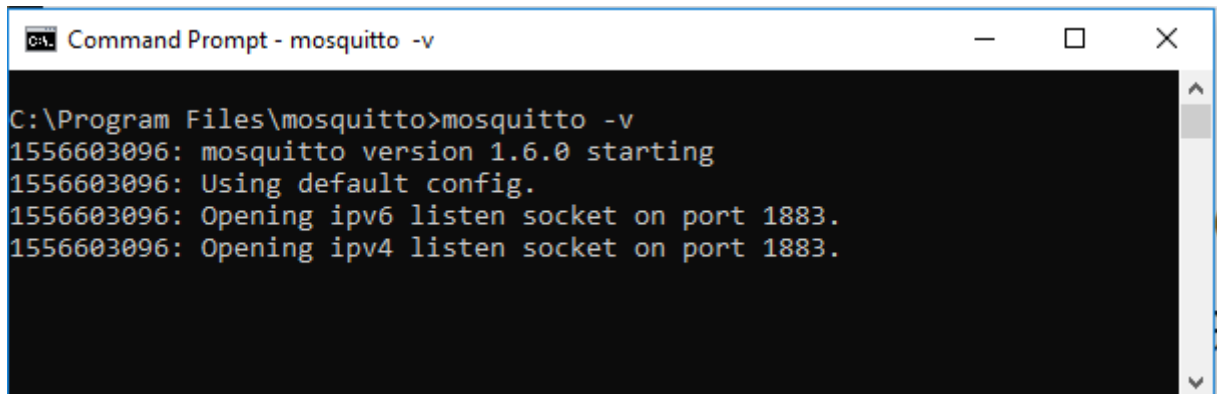


Рисунок 4.4 – Приклад відображення вихідних даних на ПК

4.5 Повідомлення

В якості повідомлень виступає інформація з екранної форми платформи OpenHAB а також результати роботи брокера MQTT.

Наприклад, на рис. 4.5 зображено приклад повідомлення про успішне налаштування роботи брокера MQTT.



```
Command Prompt - mosquitto -v

C:\Program Files\mosquitto>mosquitto -v
1556603096: mosquitto version 1.6.0 starting
1556603096: Using default config.
1556603096: Opening ipv6 listen socket on port 1883.
1556603096: Opening ipv4 listen socket on port 1883.
```

Рисунок 4.5 – Успішне налаштування роботи брокера MQTT

При помилці спілкування з МК за допомогою протоколу MQTT буде повідомлення, як на рис. 4.6.

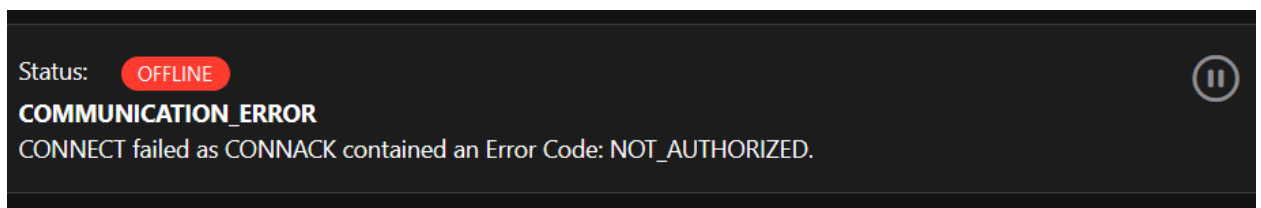


Рисунок 4.6 – Приклад успішного спілкування з МК за допомогою протоколу Modbus

5 КЕРІВНИЦТВО КОРИСТУВАЧА

5.1 Призначення програмно-апаратного забезпечення

Проведена робота та розроблені програмні та апаратні компоненти направлені конструюванні системи програмно-апаратних засобів віддаленого моніторингу електроспоживання для використання в системах домашньої автоматизації. Апаратна частина стосується в конструюванні прототипу, під'єднанні датчиків до МК, програмна – налаштування правильної роботи алгоритмів та віддаленого моніторингу даних. В результаті прототип працює стабільно та правильно у відповідності до вимог.

5.2 Умови виконання програм

Серед умов використання є наступні:

- прототип під'єднано до електромережі;
- мережа прототипу в даний момент має доступ до інтернет;
- на сервері віддаленої автоматизації не проводяться технічні роботи;
- користувач авторизований у системі.

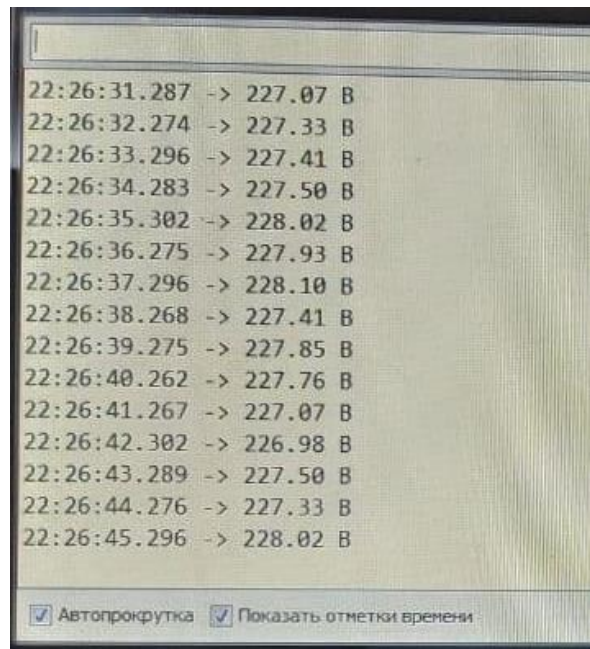
Окрім цього, для використання системи віддаленого моніторингу необхідно мати комп'ютер з швидкісним інтернет-з'єднанням, що задовольняє наступні апаратно-технічні вимоги:

- ОС Microsoft Windows 7 або вище;
- процесор з частотою 0.1 ГГц або вище;
- RAM 512 МБ або вище;
- відеокарта з підтримкою роздільної здатності екрану 1280 x 720 пікселів та вище;
- швидкісне інтернет-з'єднання від 1 Мбіт/с або вище;
- клавіатура та миша.

5.3 Виконання програм

Основна програма системи віддаленого моніторингу отримує дані з датчиків, усереднює їх та очищує від шумів, які потім конвертуються у необхідний формат за допомогою алгоритмів.

Таким чином, прототип перевіряє наявність напруги у мережі, вимірює її, а також струм, який споживає підключений електроприлад. Таким чином вираховується і потужність. На рис. 5.1 зображено приклад виведення проміжних даних, що будуть відправлені на віддалений сервер.



22:26:31.287	->	227.07 B
22:26:32.274	->	227.33 B
22:26:33.296	->	227.41 B
22:26:34.283	->	227.50 B
22:26:35.302	->	228.02 B
22:26:36.275	->	227.93 B
22:26:37.296	->	228.10 B
22:26:38.268	->	227.41 B
22:26:39.275	->	227.85 B
22:26:40.262	->	227.76 B
22:26:41.267	->	227.07 B
22:26:42.302	->	226.98 B
22:26:43.289	->	227.50 B
22:26:44.276	->	227.33 B
22:26:45.296	->	228.02 B

☒ Автопрокрутка ☒ Показати отметки времени

Рисунок 5.1 – Приклад виведення проміжних даних

5.4 Повідомлення

Повідомленнями виступає інформація з екранної форми, що може бути відображена в системі домашньої автоматизації OpenHAB.

При виникненні ситуації, що МК не буде посилати данні на віддалений сервер, наприклад при помилці живлення або поганому зв'язку, в системі OpenHAB буде виводитись значення NULL в відповідних полях. Приклад наведено на рис. 5.2 та 5.3.

G	countA String · Point GenericMQTTThing_countA	NULL >
G	countV String · Point GenericMQTTThing_countV	NULL >
G	countW String · Point GenericMQTTThing_countW	NULL >

Рисунок 5.2 – Приклад виведення нульових значень при неможливості отримати коректні дані

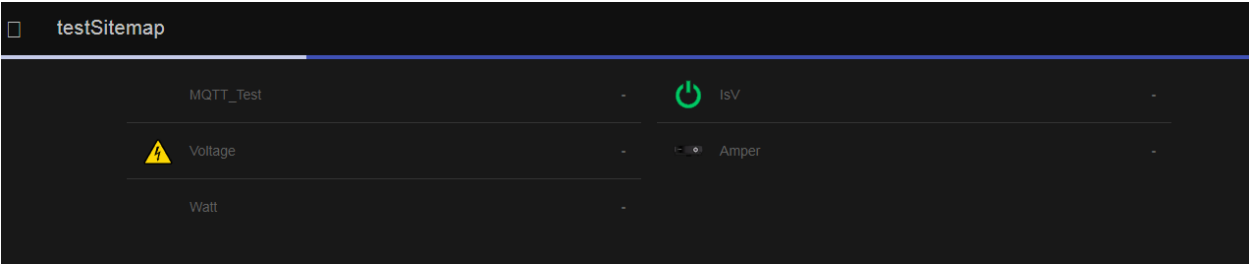


Рисунок 5.3 – Приклад виведення нульових значень при неможливості отримати коректні дані

ВИСНОВКИ

В результаті виконання дипломної роботи було проаналізовано теоретичні основи розробки систем контролю споживання електроенергії, а також досліджено програмну та апаратну складові систем віддаленого моніторингу енергоспоживання.

Розроблене програмне та апаратне забезпечення надає користувачам можливість в режимі реального часу відстежувати та аналізувати споживання електроенергії, а також реагувати на зміни.

Наукова новизна роботи полягає в тому, що запропоновані архітектурні рішення забезпечують можливість інтеграції розробленого програмно-апаратного забезпечення в будь-які популярні системи домашньої автоматизації, а також більш високу у порівнянні з аналогами точність вимірювань показників електроспоживання.

Практична цінність роботи полягає в тому, що впровадження розробленої системи віддаленого моніторингу споживання електроенергії сприятиме більш свідомому та ефективному використанню ресурсів, дозволить користувачам виявляти енерговитратні прилади, визначати періоди підвищеного споживання, а також реагувати на стан електроприладів у реальному часі. Відповідно, використання системи дозволить оптимізувати витрати на енергоспоживання та покращити загальну енергоефективність.

В майбутній роботі можливо розглянути подальшу оптимізацію програмно-апаратного забезпечення, зокрема з використанням інших методів фільтрації, протоколу передачі даних Bluetooth, більш точних датчиків для вимірювання показників енергоспоживання, що сприятиме підвищенню ефективності системи.

Результати роботи були представлені на Міжнародній НПК «Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій» (12-14 грудня 2022 року, м. Запоріжжя, Україна).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Security.org | What Is Home Automation and How Does It Work? [Electronic resource]. – Access mode: <https://www.security.org/home-automation/>.
2. Investopedia | Smart Home: Definition, How They Work, Pros and Cons. [Electronic resource]. – Access mode: <https://www.investopedia.com/terms/s/smart-home.asp>.
3. Techtarget | Smart Home. [Electronic resource]. – Access mode: <https://www.techtarget.com/iotagenda/definition/smart-home-or-building>.
4. Smartgrid | The Smart Home. [Electronic resource]. – Access mode: https://www.smartgrid.gov/the_smart_grid/smart_home.html.
5. Енергетика: історія, теперішнє та майбутнє | Сучасні автоматизовані системи контролю та обліку енергоресурсів (АСКОЕ) [Електрон. ресурс]. – Режим доступу: <https://cutt.ly/xZKDjxq>.
6. Kyivstar Business Hub | Smart metering: що це таке, навіщо та де застосовується [Електрон. ресурс]. – Режим доступу: <https://hub.kyivstar.ua/news/smart-metering-shho-cze-take-navishho-ta-de-zastosovuyetsya/>.
7. Kraton | Організація обліку теплової енергії [Електрон. ресурс]. – Режим доступу : <https://kriaton.com.ua/ua/a471553-organizatsiya-ucheta-teplovoj.html>.
8. Indiamart | Single Phase Smart Meters. [Electronic resource]. – Access mode: <https://www.indiamart.com/proddetail/single-phase-smart-meters-13091265912.html>.
9. Алло | Радіокерована розетка з лічильником енергоспоживання Ajax Socket Black [Електрон. ресурс]. – Режим доступу: [https://allo.ua/ru/umnye-rozetki/radioupravljajemaja-umnaja-rozetka-so-schetchikom-jenerGOPOTREBLenija-ajax-socket-black.html](https://allo.ua/ru/umnye-rozetki/radioupravljajemaja-umnaja-rozetka-so-schetchikom-jenerGOPOTREBlenija-ajax-socket-black.html).
10. Розетка | Розумна Wi-Fi розетка Sonoff S26 16A R2 [Електрон. ресурс]. – Режим доступу: <https://rozetka.com.ua/42343656/p42343656/>.

11. Smart energy metering and billing system / [A. Rabbani, P. Jain, P. Kumar et al.]. // International journal of engineering sciences & research technology. – 2019. – Vol. 8, № 6. – P. 98-106.
12. Circuit Digest | IoT Based Electricity Energy Meter using ESP12 and Arduino. [Electronic resource]. – Access mode: <https://circuitdigest.com/microcontroller-projects/iot-electricity-energy-meter-using-esp12-arduino>.
13. Circuit Digest | IoT Based Raspberry Pi Smart Energy Monitor. [Electronic resource]. – Access mode: <https://circuitdigest.com/microcontroller-projects/iot-based-raspberry-pi-smart-energy-meter>.
14. Kumar, V. A review on power energy efficient monitoring system / V. Kumar, S. Naveen // International journal of engineering sciences & research technology. – 2018. – Vol. 7, № 9. – P. 121-127.
15. Node-RED| Low-code programming for event-driven applications [Electronic resource]. – Access mode: <https://nodered.org>.
16. Arduino | Getting Started with Arduino products [Electronic resource]. – Access mode: <https://www.arduino.cc/en/Guide>.
17. Platformio | Professional collaborative platform for embedded development [Electronic resource]. – Access mode: <https://platformio.org>.
18. GadgetsHelp.com | Як програмувати та керувати Arduino за допомогою Python [Електрон. ресурс]. – Режим доступа: <https://gadgetshelp.com/programmirovaniye/kak-programmirovat-i-upravlyat-arduino-s-pomoshchiu-python/>.
19. Microchip | Atmel Studio [Electronic resource]. – Access mode: <https://www.microchip.com/en-us/tools-resources/develop/microchip-studio>.
20. IOTDESIGN PRO | Getting started with Lua Language on ESP32: Program ESP32 using ChiliPeppr's Web IDE [Electronic resource]. – Access mode: <https://iotdesignpro.com/projects/getting-started-with-lua-programming-on-esp32>.
21. Random Nerd Tutorials | Build an ESP8266 Web Server – Code and Schematics (NodeMCU) [Electronic resource]. – Access mode:

<https://randomnerdtutorials.com/esp8266-web-server/>.

22. Microcontrollers Lab | Accessing ESP32 web server from anywhere in the world (ESP8266 compatible) [Electronic resource]. – Access mode: <https://microcontrollerslab.com/accessing-esp32-web-server-anywhere-world-esp8266/>.

23. Random Nerd Tutorials | Telegram: Control ESP32/ESP8266 Outputs (Arduino IDE) [Electronic resource]. – Access mode: <https://randomnerdtutorials.com/telegram-control-esp32-esp8266-nodemcu-outputs/>.

24. Thingier.io Documentation | OVERVIEW [Electronic resource]. – Access mode: <https://docs.thingier.io>.

25. OpenHAB | Welcome [Electronic resource]. – Access mode: <https://www.openhab.org/docs>.

26. Techtarget | Top 12 most commonly used IoT protocols and standards [Electronic resource]. – Access mode: <https://www.techtarget.com/iotagenda/tip/Top-12-most-commonly-used-IoT-protocols-and-standards>.

27. Avsystem | Everyone is using Bluetooth Low Energy – should you? [Electronic resource]. – Access mode: <https://www.avsystem.com/blog/bluetooth-low-energy-ble/>.

28. Iotforall | What Is Cellular IoT? [Electronic resource]. – Access mode: <https://www.ietfforall.com/what-is-cellular-iot>.

29. What is MQTT? | Amazon Web Services? [Electronic resource]. – Access mode: <https://aws.amazon.com/what-is/mqtt/>.

30. HiveMQ | MQTT Vs. HTTP for IoT [Electronic resource]. – Access mode: <https://www.hivemq.com/blog/mqtt-vs-http-protocols-in-iot-iiot/>.

31. ReadWrite | Websockets and IoT: Why don't the two go together? [Electronic resource]. – Access mode: <https://readwrite.com/websockets-iot-two-dont-go-together/>.

32. Arduino Store | Arduino Uno Rev3 [Electronic resource]. – Access

mode: <http://store.arduino.cc/products/arduino-uno-rev3>.

33. Fabacademy | ESP8266 Introduction [Electronic resource]. – Access mode: <http://fabacademy.org/archives/2015/doc/networking-esp8266.html>.

34. Arduino.ua | Wi-Fi модуль LuaNode32 з ESP-32 [Електрон. Ресурс]. – Режим доступа: <https://arduino.ua/prod2041-Wi-Fi-modyl-luanode32-s-esp-32>.

35. M5Stack | ATOM Lite ESP32 IoT Development Kit [Electronic resource]. – Access mode: <https://shop.m5stack.com/products/atom-lite-esp32-development-kit>.

36. RoboticX | HLK-PM01 AC-DC Converter 220V TO 5V 3W power module [Electronic resource]. – Access mode: <https://roboticx.ps/product/hlk-pm01-ac-dc-converter-220v-to-5v-3w-mini-power-supply-module>.

37. Quartz Components | 5V 10A Relay Module [Electronic resource]. – Access mode: <https://quartzcomponents.com/products/5v-10a-relay-module>.

38. Free Delivery | Датчик контролю наявності напруги 220В з опторозв'язкою [Електрон. ресурс]. – Режим доступа: <https://freedelivery.com.ua/arduino-100/datchiki-130/datchik-kontrolja-nalichija-setevogo-naprijazhenija-220v-s-optorazvjazkoj.html>.

39. Ebay | LC Technology AC Voltage Sensor Module Zmpt 101b 250V for Arduino Workshop 5-30V [Electronic resource]. – Access mode: <https://www.ebay.com/itm/401752805821>.

40. Arduino.ua | Датчик струму ACS712 30A GY-712 [Електрон. ресурс]. – Режим доступа: <https://arduino.ua/prod618-datchik-toka-ac712-30a-gy-712>.

41. Siqma Robotics | ZMCT103C Current Transformer [Electronic resource]. – Access mode: <https://store.siqma.com/ZMCT103C-Current-Transformer.html>.

42. Arduino.ua | Вимірювання сили струму датчиком SCT-013-030 [Електрон. ресурс]. – Режим доступа: <https://arduino.ua/art65-izmerenie-sili-toka-datchikom-sct-013-030>.

ДОДАТОК А
Текст програми

```

#include <PubSubClient.h>

#include <WiFi.h>

// ampere sensor pin
#define AMPERE_SENSOR_PIN 33
// voltage sensor pin
#define VOLTAGE_SENSOR_PIN 32
// voltage availability pin
#define VOLTAGE_AVAILABILITY_PIN 21
// relay pin
#define RELAY_PIN = 25

// filter element count
const int FILTER_ELEM_COUNT = 30;
// ADC max
const float MAX_ANALOG_READ = 4095.0;
// MAX analog read voltage
const float MAX_ANALOG_READ_V = 3.3;
// count of coil turn
const int COUNTS_OF_TURN = 2500;
// resistor (Ohm)
const int RESISTANS = 39;
// MAX coil ampere
const int MAX_AMPESR_COIL = 70;

uint32_t mainTimer = 0;

int isTurnRelay = 1;

// your WIFI SSID and password
const char *WIFI_SSID_NAME = "YOUR_WIFI_SSID_NAME";
const char *WIFI_PASSWORD = "YOUR_WIFI_PASSWORD";

// your MQTT server adress
const char *MQTT_SERVER = "YOUR_MQTT_SERVER";

```

```

// your MQTT server port
const int MQTT_PORT = 1883;

// topics names
const String TOPIC_NAME_SYNC_T = "monitor/time";
const String TOPIC_NAME_IS_V = "monitor/isV";
const String TOPIC_NAME_A = "monitor/A";
const String TOPIC_NAME_V = "monitor/V";
const String TOPIC_NAME_W = "monitor/W";
const String TOPIC_NAME_RELAY = "monitor/isTurnRelay";

WiFiClient espClient;
PubSubClient client(espClient);

void setup()
{
    Serial.begin(9600);

    // start connection via MQTT
    setup_wifi();

    client.setServer(MQTT_SERVER, MQTT_PORT);
    client.setCallback(callback);

    pinMode(RELAY_PIN, OUTPUT);
    changeRelayState();
}

void loop()
{
    // getting main value
    double currentAmpereValue = getAmpereValue();
    double currentVoltageValue = getVoltageValue();
    bool voltageAvailability = isAvailabilityVoltage();

```

```

// print and send value every second
if (millis() - mainTimer > 1000)
{
    serialPrint(currentVoltageValue, currentAmpereValue,
voltageAvailability);

    sendMessageMQTT(currentVoltageValue, currentAmpereValue,
voltageAvailability);

    changeRelayState();

    mainTimer = millis();
}
}

// function change state of relay
void changeRelayState()
{
    if (isTurnRelay == 1)
    {
        Serial.println("Turning ON relay");
        digitalWrite(RELAY_PIN, HIGH);
    }
    else
    {
        Serial.println("Turning OFF relay");
        digitalWrite(RELAY_PIN, LOW);
    }
}

// function getting voltage value
double getVoltageValue()
{
    int sensor_Value = 0;
    float v_Out = 0.0;

```

```

static uint32_t timerV = 0;

// MIN and MAX from ADC
static float Vmin = 1024;
static float Vmax = 0;

static float result_V = 0;

// read the analog in value
sensor_Value = analogRead(VOLTAGE_SENSOR_PIN);
// convert the read value into voltage
v_Out = 250 * (sensor_Value / MAX_ANALOG_READ);

// finding the resulting maximum and minimum
if (v_Out > Vmax)
{
    Vmax = v_Out;
}

if (v_Out < Vmin)
{
    Vmin = v_Out;
}

// update the result every 1/10 second
if (millis() > timerV + 100)
{
    timerV = millis();

    // calculate the actual voltage value taking into account the root of
    two, that is, without peak voltage
    float result = ((Vmax - Vmin) * 2) / sqrt(2);

    if (Vmin > 0)
    {
        result = 0;
    }
}

```

```

    }

    Vmin = MAX_ANALOG_READ;
    Vmax = 0;

    // voltage filtering to avoid surges
    result_V = runMiddleArifmOptimV(result);
}

return (result_V);
}

// function getting ampere value
double getAmpereValue()
{
    static uint32_t timerA1 = 0;
    static uint32_t timerA2 = 0;

    static int maxICur = 0;
    static int maxI = 0;

    static float result_I = 0;

    // every millisecond we read the value and find the maximum
    if ((millis() - timerA1) > 1)
    {
        timerA1 = millis();
        int readValue = analogRead(AMPERE_SENSOR_PIN);
        if (readValue > maxICur)
            maxICur = readValue;
    }

    // every 1/10 second we process the received maximum value
    if ((millis() - timerA2) > 100)
    {
        timerA2 = millis();
        maxI = maxICur;
    }
}

```

```

    maxICur = 0;

    float res = ((maxI - MAX_ANALOG_READ / 2.0) * MAX_ANALOG_READ_V) /
MAX_ANALOG_READ;
    float nCurrThruResistorPP = (res / RESISTANS) * COUNTS_OF_TURN;
    float nCurrThruResistorRMS = nCurrThruResistorPP * 0.707;

    if (nCurrThruResistorRMS < 0)
    {
        nCurrThruResistorRMS = 0;
    }

    // process the result with a filter
    result_I = runMiddleArifmOptim(nCurrThruResistorRMS);
}
return (result_I);
}

// function getting availability voltage
float isAvailabilityVoltage()
{
    int valuePinIsV = digitalRead(VOLTAGE_AVAILABILITY_PIN);
    bool resultIsVoltage;

    if (valuePinIsV)
    {
        resultIsVoltage = true;
    }
    else
    {
        resultIsVoltage = false;
    }

    return resultIsVoltage;
}

```

```

// optimal arithmetic mean for current strength
float runMiddleArifmOptim(float filterValueA)
{
    static int t = 0;
    static float vals[FILTER_ELEM_COUNT];
    static float average = 0;

    if (++t >= FILTER_ELEM_COUNT)
    {
        t = 0;
    }

    average -= vals[t];
    average += filterValueA;
    vals[t] = filterValueA;

    float result = (float)average / FILTER_ELEM_COUNT;

    return (result);
}

// optimal running arithmetic mean for voltage
float runMiddleArifmOptimV(float filterValueV)
{
    static int t2 = 0;
    static float vals2[FILTER_ELEM_COUNT];
    static float average2 = 0;

    if (++t2 >= FILTER_ELEM_COUNT)
    {
        t2 = 0;
    }

    average2 -= vals2[t2];

```

```

average2 += filterValueV;
vals2[t2] = filterValueV;

float result = (float)average2 / FILTER_ELEM_COUNT;

return (result);
}

// function for sending messages via MQTT
void sendMessageMQTT(float valueV, float valueI, bool voltageAvailability)
{
    // if you are not connected to the server, then reconnect
    if (!client.connected())
    {
        reconnect();
    }

    // if the server is not running, a connection occurs
    if (!client.loop())
    {
        client.connect("ESP8266Client");
    }

    // getting power value
    float powerValue = valueI * valueV;

    // getting String value
    String synchTimeMessage = "Time = " + String(millis());
    String isV = "";
    String valueV_str = String(valueV) + " V";
    String valueI_str = String(valueI) + " A";
    String valueW_str = String(powerValue) + " W";

    if (voltageAvailability)
    {

```

```

        isV = "+";
    }
    else
    {
        isV = "-";
    }

    // sending messages to relevant topics
    client.publish(TOPIC_NAME_SYNC_T, synchTimeMessage.c_str());
    client.publish(TOPIC_NAME_IS_V, isV.c_str());
    client.publish(TOPIC_NAME_V, valueV_str.c_str());
    client.publish(TOPIC_NAME_A, valueI_str.c_str());
    client.publish(TOPIC_NAME_W, valueW_str.c_str());
}

// WiFi launch function
void setup_wifi()
{
    delay(10);

    // We start by connecting to a WiFi network
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(WIFI_SSID_NAME);

    WiFi.begin(WIFI_SSID_NAME, WIFI_PASSWORD);

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

    Serial.println("");
    Serial.print("WiFi connected - ESP IP address: ");

```

```

    Serial.println(WiFi.localIP());
}

// function for receiving a message from a topic
void callback(String topic, byte *message, unsigned int length)
{
    Serial.print("Message arrived on topic: ");
    Serial.print(topic);
    Serial.print(". Message: ");

    String messageTemp;

    for (int i = 0; i < length; i++)
    {
        Serial.print((char)message[i]);
        messageTemp += (char)message[i];
    }

    Serial.println();

    if (messageTemp == "+")
    {
        isTurnRelay = 1;
        Serial.println("Turn On");
    }
    else if (messageTemp == "-")
    {
        isTurnRelay = 0;
        Serial.println("Turn Off");
    }
    else
    {
        Serial.println("Error Message");
    }
}

```

```

// MQTT restart function
void reconnect()
{
    // Loop until we're reconnected
    while (!client.connected())
    {
        Serial.print("Attempting MQTT connection...");

        if (client.connect("ESP8266Client"))
        {
            Serial.println("connected");
            client.subscribe(TOPIC_NAME_RELAY);
        }
        else
        {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 5 seconds");

            // Wait 5 seconds before retrying
            delay(5000);
        }
    }
}

// function sending data to serial port
void serialPrint(float valueV, float valueI, bool voltageAvailability)
{
    float powerValue = valueI * valueV;

    String isV = "";
    String valueV_str = String(valueV) + " V";
    String valueI_str = String(valueI) + " A";
    String valueW_str = String(powerValue) + " W";

```

```
String synchTime = "Time" + String(millis());

if (voltageAvailability)
{
    isV = "Turn on";
}
else
{
    isV = "Turn off";
}

Serial.println(synchTime);
Serial.println(isV);
Serial.println(valueV_str);
Serial.println(valueI_str);
Serial.println(valueW_str);
}
```

ДОДАТОК Б
Слайди презентації

Національний університет «Запорізька політехніка»

Кафедра програмних засобів

ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ВІДДАЛЕНОГО МОНІТОРИНГУ СПОЖИВАННЯ ЕЛЕКТРОЕНЕРГІЇ

Виконав
студент групи КНТ-212м

Євгеній ТРОЦ

Керівник
к.т.н. доцент

Анжеліка ПАРХОМЕНКО

Рисунок Б.1 – Слайд №1

Актуальність роботи



Приклад використання
Телеграм



Панель моніторингу системи thinger.io



Сторінка моніторингу системи



Приклад панелі openHAB

Рисунок Б.2 – Слайд №2

Мета та задачі дипломної роботи

Мета роботи – дослідження та розробка системи віддаленого моніторингу споживання електроенергії для контролю процесу енергоспоживання, підвищення ефективності споживання електроенергії та безпеки при використанні електроприладів.

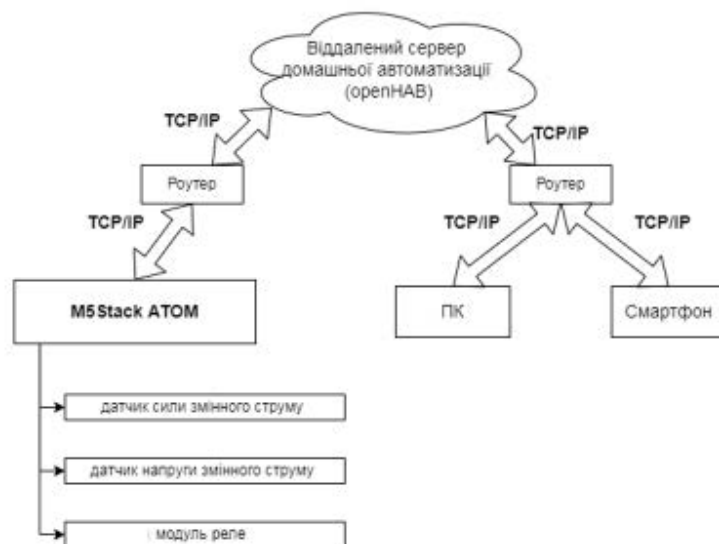
Задачі роботи:

- дослідження та вибір технологій та засобів прототипування системи віддаленого моніторингу споживання електроенергії;
- розробка та тестування апаратних та програмних засобів системи;
- розробка програмної документації.

3

Рисунок Б.3 – Слайд №3

Архітектура системи віддаленого моніторингу



4

Рисунок Б.4 – Слайд №4

Порівняльний аналіз особливостей протоколів передачі даних

Характеристика	WebSocket	Стільниковий зв'язок	Bluetooth / BLE	MQTT	HTTP
Діапазон	локальна мережа, Інтернет	глобальна мережа	локальна мережа, ближня зона	локальна мережа, глобальна мережа	глобальна мережа
Швидкість	висока	середня	низька	висока	середня
Затримка	низька	висока	середня	низька	середня
Надійність	висока	середня	висока	висока	середня
Безпека	висока	середня	середня	висока	середня
Економічність	середня	висока	середня	висока	середня
Застосування	онлайн-ігри, відеоконференції, push-повідомлення	передача даних, голосовий зв'язок, відеозв'язок	бездротові пристрої, сенсорні мережі	IoT, промислове обладнання	веб-додатки, передача даних
Недоліки	великий рівень енергоспоживання	висока вартість, можливість обмежень	малий радіус дії	більш складна реалізація	незахищеність даних

5

Рисунок Б.5 – Слайд №5

Порівняльний аналіз засобів моніторингу та керування

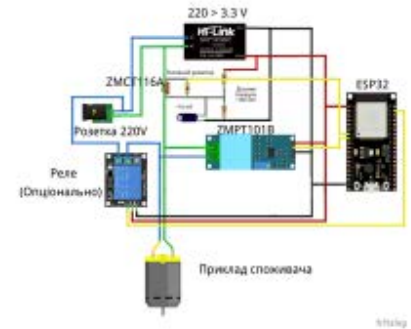
Назва	Переваги	Недоліки
Веб-сторінка	Високий рівень кастомізації інтерфейсу, унікальність дизайну, можливість локального використання	Необхідність створювати інтерфейс з нуля, відсутність готових інструментів для створення інтерфейсу
Telegram	Зручність інтеграції з популярним месенджером, відносна простота	Можливість взаємодії лише текстовими командами, низька наочність результатів моніторингу, неможливість локального використання
Thingier.io	Наявність готового хмарного сервісу, наявність готових компонентів інтерфейсу	Погана інтеграція з багатьма компонентами IoT, неможливість локального використання
Панель openHAB	Наявність готових компонентів інтерфейсу, гарна інтеграція з багатьма компонентами IoT, можливість локального використання	Погана документація, неможливість повної кастомізації інтерфейсу

6

Рисунок Б.6 – Слайд №6

Апаратне забезпечення системи віддаленого моніторингу електрозбереження

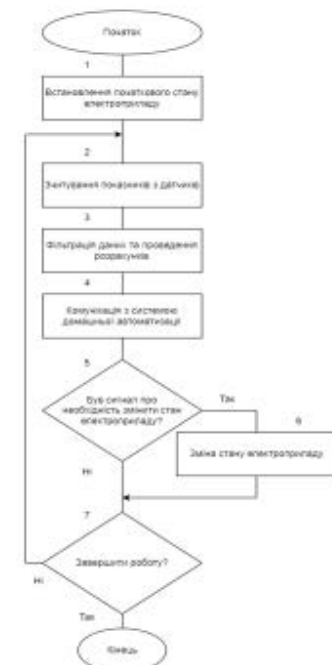
Компонент	Функція
Датчик напруги змінного струму	Зчитування показань напруги змінного струму в мережі. Використовується також для обчислення потужності прилада споживача
Модуль електромеханічного реле	Віддалений контроль стану ввімкнення/вимкнення електроприладу
Датчик сили струму	Зчитування показань сили струму, яку використовує споживач. Використовується також для обчислення потужності.
Мікроконтролер	Забезпечення отримання даних з датчиків та відправка їх на віддалений сервер.
Модуль перетворювача напруги	Пониження мережевої напруги для можливості роботи системи від електромережі



7

Рисунок Б.7 – Слайд №7

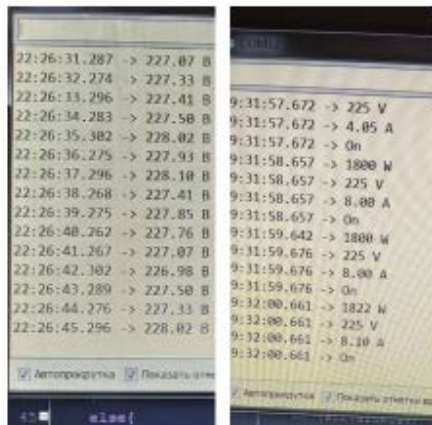
Алгоритм функціонування системи віддаленого моніторингу



8

Рисунок Б.8 – Слайд №8

Результати тестування



Приклади отримання тестових значень через послідовний порт

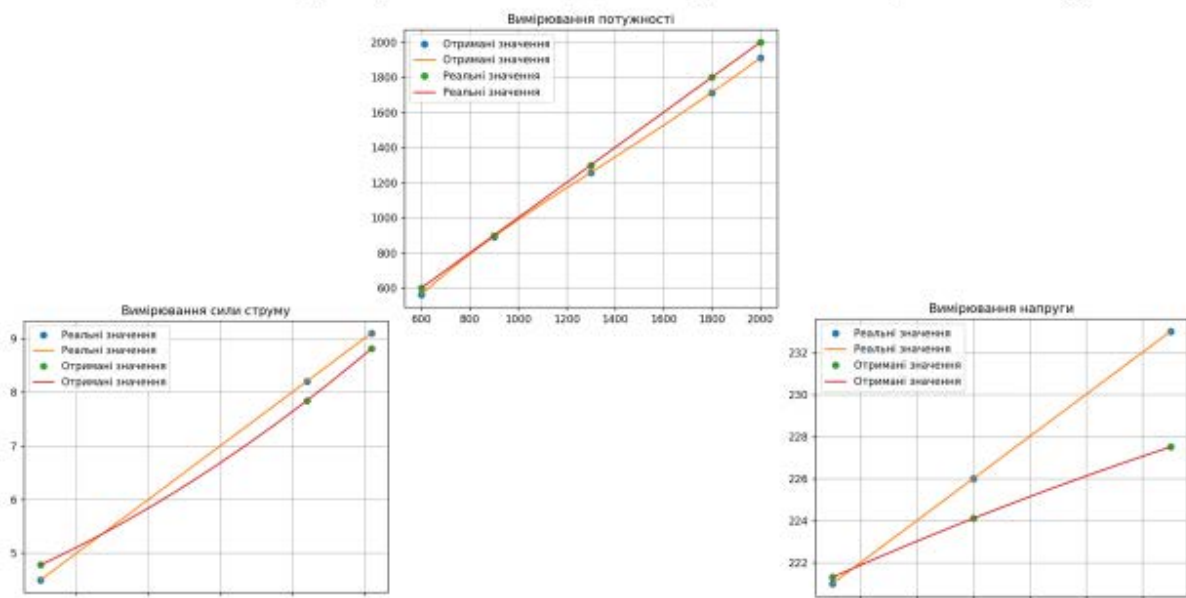
Реальне значення, Вт	Отримане значення	
	Значення, Вт	Похибка, %
600	563	6,2
900	893	0,8
1300	1258	3,2
1800	1714	4,8
2000	1910	4,5
Реальне значення, А	Середнє значення, А	Похибка, %
4,5	4,78	6,2
8,2	7,84	4,4
9,1	8,81	3,2
Реальне значення, В	Середнє значення, В	Похибка, %
221	221,3	0,1
226	224,1	0,8
233	227,5	2,4

Таблиці результатів тестувань (значення виміряні в різні часові проміжки і на різних ділянках електромережі)

9

Рисунок Б.9 – Слайд №9

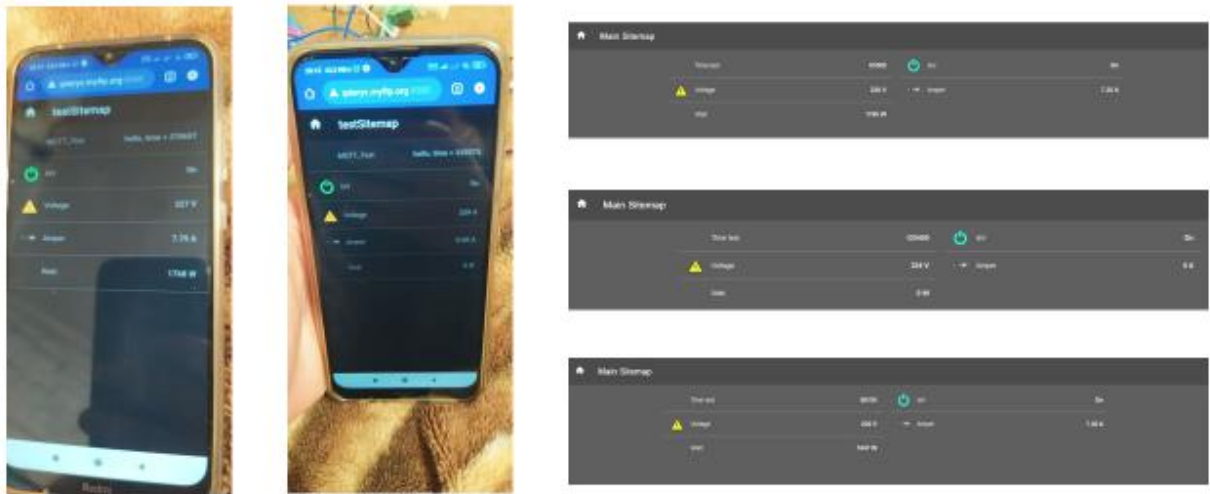
Аналіз результатів дослідження прототипу



10

Рисунок Б.10 – Слайд №10

Результати роботи прототипу системи віддаленого моніторингу на сервері openHAB



11

Рисунок Б.11 – Слайд №11

Висновки

- Наукова новизна роботи полягає в тому, що запропоновані архітектурні рішення забезпечують можливість інтеграції розробленого програмно-апаратного забезпечення в будь-які популярні системи домашньої автоматизації, а також більш високу у порівнянні з аналогами точність вимірювань показників електроспоживання.
- Практична цінність роботи полягає в тому, що впровадження розробленої системи віддаленого моніторингу споживання електроенергії сприятиме більш свідомому та ефективному використанню ресурсів, дозволить користувачам виявляти енергозатратні прилади, визначати періоди підвищеного споживання, а також реагувати на стан електроприладів у реальному часі. Відповідно, використання системи дозволить оптимізувати витрати на енергоспоживання та покращити загальну енергоефективність.
- В майбутній роботі можливо розглянути подальшу оптимізацію програмно-апаратного забезпечення, зокрема з використанням інших методів фільтрації, протоколу передачі даних Bluetooth, більш точних датчиків для вимірювання показників енергоспоживання, що сприятиме підвищенню ефективності системи.
- Результати роботи були представлені на Міжнародній НПК «Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій» (12-14 грудня 2022 року, м. Запоріжжя, Україна).

12

Рисунок Б.12 – Слайд №12