ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «СИСТЕМА АВТОМАТИЗАЦІЇ ПРОМИСЛОВИХ ПРОЦЕСІВ НА БАЗІ МІКРОПРОЦЕСОРНИХ СИСТЕМ З ІНТЕРНЕТОМ РЕЧЕЙ»

на здобуття освітнього ступеня магістра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Андрій ЗАЯЧКОВСЬКИЙ*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувачі*

Виконав:
здобувач вищої освіти Андрій ЗАЯЧКОВСЬКИЙ
група ІСДМ-64

Керівник:
науковий ступінь, вчене звання Ігор СІНІЦИН
д.т.н., професор

Рецензент:
науковий ступінь, вчене звання Вікторія ЖЕБКА
Ім'я, ПРІЗВИЩЕ

Київ – 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Інформаційних систем та технологій

Ступінь вищої освіти Магістр

Спеціальність Інформаційні системи та технології

Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ

_____Каміла СТОРЧАК
«____» _____20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ СТУДЕНТУ
Заячковському Андрію Володимировичу
(прізвище, ім'я, по батькові здобувача)

1.Тема кваліфікаційної роботи: «Система автоматизації промислових процесів на базі мікропроцесорних систем з Інтернетом речей»

керівник кваліфікаційної роботи: Ігор СІНІЦИН д.т.н., професор
(ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від «15» жовтня 2024 року № 320.

2. Строк подання кваліфікаційної роботи: 27 грудня 2024 року.

3. Вихідні дані до кваліфікаційної роботи: Список апаратних компонентів для реалізації інтегрованої IoT-системи: контролери, сенсори, мікропроцесорні платформи;
Протоколи зв'язку для забезпечення передачі даних: MQTT, CoAP, HTTP(S);
Інструменти для збору, аналізу та візуалізації даних (Node-RED, AWS IoT, Azure IoT);
Експериментальні дані з використанням сенсорів і контролерів для перевірки концепції;
науково-технічна література з питань Інтернету речей та мікропроцесорних систем.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Аналіз сучасних технологій автоматизації промислових процесів з використанням IoT та мікропроцесорних систем.
2. Розробка архітектури інтегрованої IoT-системи, вибір апаратних і програмних компонентів.
3. Реалізація та тестування прототипу системи автоматизації промислового процесу з використанням IoT та оцінка її ефективності.

5. Перелік ілюстративного матеріалу: *презентація*
Представлення дослідження
Завдання представлені перед дослідженням
Об'єкт, предмет та мета дослідження
Аналіз існуючих рішень у сфері автоматизації промислових процесів та IoT
Розробка концепції автоматизації
Проєктування апаратної частини
Розробка програмного забезпечення
Зображення компонентів системи
Висновки

6. Дата видачі завдання: 15 жовтня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	15.10 – 23.10.24	
2	Розробка архітектури інтегрованої IoT-системи	24.10 – 01.11.24	
3	Вибір апаратних і програмних компонентів системи	02.11 – 07.11.24	
4	Розробка алгоритмів збору, обробки та передачі даних	08.11 – 16.11.24	
5	Тестування прототипу IoT-системи в умовах, наближених до реальних	17.11 – 21.11.24	
6	Аналіз результатів тестування та підготовка висновків	22.11 – 11.12.24	
7	Оформлення роботи: вступ, висновки, реферат	12.12 – 20.12.24	
8	Розробка демонстраційних матеріалів	21.12 – 25.12.24	

Здобувач вищої освіти _____ Андрій ЗАЯЧКОВСЬКИЙ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник роботи
кваліфікаційної роботи _____ Ігор СІНІЦИН
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 74 стор., 23 рис., 4 табл., 30 джерел.

Мета роботи – дослідження методів інтеграції мікропроцесорних систем з Інтернетом речей для автоматизації промислових процесів, створення прототипу інтегрованої IoT-системи та оцінка її ефективності.

Об'єкт дослідження – промислові процеси, які можуть бути автоматизовані за допомогою інтегрованих IoT та мікропроцесорних систем.

Предмет дослідження – методи і засоби інтеграції мікропроцесорних систем з технологіями Інтернету речей, алгоритми управління та аналізу даних для підвищення ефективності автоматизованих систем.

Короткий зміст роботи:

Робота зосереджена на вивченні можливостей використання Інтернету речей у поєднанні з мікропроцесорними системами для автоматизації промислових процесів. Проведено аналіз сучасних технологій, розроблено архітектуру інтегрованої IoT-системи, вибрано апаратні та програмні компоненти, створено та протестовано прототип інтегрованої системи. Особливу увагу приділено забезпеченню ефективності, надійності та безпеки роботи IoT-рішень у промислових умовах.

КЛЮЧОВІ СЛОВА: ІНТЕРНЕТ РЕЧЕЙ, МІКРОПРОЦЕСОРНІ СИСТЕМИ, АВТОМАТИЗАЦІЯ ПРОМИСЛОВИХ ПРОЦЕСІВ, ІОТ-СИСТЕМИ, ПРОТОКОЛИ ЗВ’ЯЗКУ, ПРОМИСЛОВІ СЕНСОРИ, АНАЛІТИКА ДАНИХ.

ABSTRACT

Text part of the master`s qualification work: 74 pages, 23 pictures, 4 table, 30 sources.

The purpose of the work to study methods for integrating microprocessor systems with the Internet of Things (IoT) for industrial process automation, develop a prototype of an integrated IoT system, and evaluate its efficiency

Object of re*search are industrial processes that can be automated using integrated IoT and microprocessor systems.*

Subject of research methods and tools for integrating microprocessor systems with IoT technologies, control algorithms, and data analysis to enhance the efficiency of automated systems.

Summary of the work:

The work focuses on exploring the potential of combining IoT with microprocessor systems to automate industrial processes. It includes an analysis of current technologies, the development of an integrated IoT system architecture, the selection of hardware and software components, and the creation and testing of a system prototype. Particular attention is paid to ensuring the efficiency, reliability, and security of IoT solutions in industrial environments.

KEYWORDS: INTERNET OF THINGS, MICROPROCESSOR SYSTEMS, INDUSTRIAL PROCESS AUTOMATION, IOT SYSTEMS, COMMUNICATION PROTOCOLS, INDUSTRIAL SENSORS, DATA ANALYTICS.

ЗМІСТ

ВСТУП

Сучасні технологічні досягнення відкривають нові шляхи для автоматизації промислових процесів, що є ключовим фактором підвищення продуктивності та ефективності виробництва. Інтернет речей (IoT) та мікропроцесорні системи є основними технологіями, що сприяють цьому розвитку. Оптимізація виробничих процесів досягається за рахунок розробки інтелектуальних систем, які здатні автоматично збирати, аналізувати та обробляти дані в режимі реального часу.

Зростаюча потреба в автоматизації промислових процесів для підвищення конкурентоспроможності робить тему *дослідження актуальною*. Використання IoT разом із мікропроцесорними системами дозволяє значно знизити виробничі витрати, підвищити якість продукції та забезпечити гнучкість процесів.

Метою магістерської роботи є дослідження методів інтеграції мікропроцесорних систем з Інтернетом речей для автоматизації промислових процесів. У роботі розглядаються сучасні підходи до розробки промислових IoT-рішень, вибір апаратних і програмних засобів для побудови таких систем, а також створення прототипу інтегрованої системи для автоматизації на базі IoT. Дослідження також охоплює питання безпеки, надійності та ефективності роботи таких рішень у реальних умовах.

Дане дослідження має на меті розширити розуміння про те, як використовувати технології IoT та мікропроцесорних систем для створення ефективних, надійних і безпечних автоматизаційних рішень у промислових процесах.

Для досягнення цієї мети необхідно вирішити *наступні завдання*:

- провести аналіз сучасного стану технологій IoT та мікропроцесорних систем у контексті автоматизації промислових процесів;
- розробити архітектуру інтегрованої IoT-системи для автоматизації промислових процесів, включаючи вибір компонентів та протоколів зв'язку;

- реалізувати прототип інтегрованої системи та протестувати її в умовах, наближених до реальних виробничих процесів;
- провести оцінку ефективності розробленої системи та порівняти її з існуючими рішеннями для автоматизації на базі IoT та мікропроцесорів.

Об'єкт дослідження: промислові процеси, що можуть бути автоматизовані за допомогою інтегрованих IoT та мікропроцесорних систем.

Предмет дослідження методи та засоби інтеграції мікропроцесорних систем з технологіями Інтернету речей для автоматизації промислових процесів, а також алгоритми управління та аналізу даних для підвищення ефективності автоматизованої системи.

До *методів дослідження* відносяться аналіз літературних джерел, пов'язаних з Інтернетом речей та мікропроцесорними системами для автоматизації промислових процесів, проєктування та моделювання інтегрованої IoT-системи, а також експериментальні дослідження і аналіз результатів роботи прототипу системи в умовах, наближених до реальних виробничих процесів.

Основними *джерелами дослідження* є наукова література з тематики Інтернету речей, аналітичні звіти з впровадження IoT-технологій в промисловості, дані з промислових сенсорів, результати технічних опитувань, експертні оцінки та інша інформація, що стосується застосування IoT для автоматизації промислових процесів.

Наукова новизна роботи полягає у розробці та обґрунтуванні практичних стратегій і рекомендацій щодо вдосконалення застосування технологій Інтернету речей в інтеграції з мікропроцесорними системами для автоматизації промислових процесів, що становить суттєвий внесок у галузь автоматизації та промислових технологій, розширюючи можливості застосування IoT для підвищення ефективності і надійності виробничих процесів.

Апробація дослідження здійснювалась у формі участі в II всеукраїнській науково-технічній конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і

світу», а також представленням статті у загальногалузевому науково-виробничому журналі «Зв’язок»

Структура роботи. Загальний обсяг роботи складає 75 сторінок друкованого тексту. Робота включає вступ, чотири розділи, висновки та список використаної літератури, який містить 30 джерел. Текст роботи містить 23 рисунків і 4 таблиці. Дана робота стане корисною для практичного впровадження в галузі інтеграції мікропроцесорних систем з Інтернетом речей (IoT) для автоматизації промислових процесів.

РОЗДІЛ 1 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ У СФЕРІ АВТОМАТИЗАЦІЇ ПРОМИСЛОВИХ ПРОЦЕСІВ ТА IoT

1.1 Огляд наявних розробок і технологій у сфері автоматизації промислових процесів та IoT

Автоматизація промислових процесів та Інтернет речей (IoT) значно змінюють традиційні підходи до управління виробничими лініями, обслуговування обладнання, збирання даних та аналізу. Вони забезпечують підвищення ефективності, зниження витрат і поліпшення якості продукції.

Автоматизація промислових процесів є ключовим фактором підвищення ефективності та продуктивності виробництва. Традиційні системи автоматизації, такі як SCADA, PLC та DCS, забезпечують надійне управління і моніторинг технологічних процесів, знижуючи вплив людського фактору та підвищуючи точність і швидкість реакції на зміни. Кожна з цих систем має свої унікальні характеристики та сфери застосування, а саме:

SCADA-системи − це комплекс апаратного та програмного забезпечення, призначений для моніторингу та управління промисловими процесами в реальному часі. Основні функції SCADA включають:

- отримує інформацію від польових пристроїв (сенсорів, вимірювальних приладів, контролерів), тобто збір даних;
- системи дозволяють візуалізувати процеси на графічних інтерфейсах, що надає операторам повну картину роботи обладнання;
- отримані дані аналізуються для виявлення відхилень від нормального функціонування;
- оператори можуть втручатися в процеси, змінювати налаштування або зупиняти обладнання в разі необхідності [1,12].

Основні компоненти SCADA:

- польові пристрої (сенсори, вимірювальні прилади, що збирають дані з об’єктів);

- комунікаційна інфраструктура (канали зв'язку, які передають дані від польових пристроїв до центрального вузла);
- центральний вузол (комп'ютери або сервери, де відбувається обробка та зберігання даних);
- інтерфейс користувача (графічні екрани, де оператори можуть взаємодіяти із системою).

PLC (Programmable Logic Controllers) − це програмовані логічні контролери, які виконують роль основного управлінського пристрою у виробничих процесах. Вони відповідають за автоматичне керування машинами та технологічними лініями.

Основні характеристики PLC:

PLC розроблені для роботи у важких промислових умовах − висока температура, вологість, пил.

Контролери програмуються за допомогою мов логічного програмування, таких як Ladder Logic або Function Block Diagram.

Можливість швидко змінювати та адаптувати програми під нові задачі.

Миттєве реагування на зміни в процесах, що дозволяє швидко виконувати управлінські рішення.

Основні функції PLC:

- управління виробничими процесами, тобто PLC виконує функції запуску, зупинки, регулювання параметрів;
- збір та обробка сигналів від польових пристроїв (кнопок, перемикачів, датчиків);
- PLC відстежує стан процесу та передає інформацію на SCADA-системи або операторські панелі [2,22].

DCS (Distributed Control Systems) − це розподілені системи управління, які забезпечують автоматизацію великих виробничих процесів, що складаються з кількох підсистем. Вони відрізняються від SCADA своєю децентралізованою структурою управління.

Основні особливості DCS:

– управлінські функції розподілені між кількома контролерами або підсистемами;
– DCS об’єднує різні частини виробничого процесу в єдину систему;
– висока стійкість до відмов завдяки розподіленій архітектурі;
– легке додавання нових модулів і розширення системи.

Основні компоненти DCS:

– локальні контролери, які керують окремими частинами виробничого процесу;
– графічні панелі для взаємодії з системою;
– інфраструктура для передачі даних між різними компонентами DCS;
– виконує функції координації та зберігання даних [3,30].

Для зручності порівняння основних характеристик та сфер застосування систем SCADA, PLC та DCS, їхні ключові параметри узагальнено в табл. 1.1.

Таблиця 1.1

Порівняння SCADA, PLC та DCS

Параметр	**SCADA**	**PLC**	**DCS**
Функція	Моніторинг і управління	Локальне управління	Розподілене управління
Архітектура	Централізована	Локальна	Децентралізована
Сфера застосування	Великі та складні системи	Окремі машини та процеси	Великі інтегровані системи
Надійність	Середня	Висока	Дуже висока
Складність	Висока	Середня	Дуже висока

Інтернет речей (IoT) відкриває нові горизонти для автоматизації промислових процесів, забезпечуючи безперервний збір і обробку даних у реальному часі. Завдяки інтеграції IoT-платформ, сенсорів і аналітичних інструментів, підприємства можуть підвищити ефективність, передбачати несправності та

оптимізувати виробництво. Далі розглянемо ключові компоненти та можливості IoT у промисловості [5,16].

Основні функції IoT-платформ:

- платформи збирають дані з різноманітних сенсорів та пристроїв, встановлених на обладнанні;
- використання алгоритмів машинного навчання та аналітики для виявлення трендів, аномалій та оптимізації процесів;
- віддалене керування обладнанням в реальному часі;
- можливість взаємодії з ERP, SCADA та іншими системами для створення комплексного середовища автоматизації [6,18].

Сенсори є ключовим елементом Інтернету речей (IoT), адже вони забезпечують зв'язок між фізичним світом і цифровими системами. Завдяки сенсорам, IoT-системи можуть збирати дані про різні параметри навколишнього середовища або стан обладнання, що дозволяє оперативно реагувати на зміни, аналізувати процеси та приймати обґрунтовані рішення для оптимізації роботи.

Типи сенсорів:

- Температурні сенсори вимірюють температуру обладнання або навколишнього середовища. Наприклад, у харчовій промисловості температурні сенсори використовуються для контролю температури в холодильних установках, забезпечуючи правильні умови зберігання продуктів.
- Вібраційні сенсори виявляють аномалії у вібрації машин, що можуть свідчити про несправності. Наприклад, у металургії вібраційні сенсори застосовуються для моніторингу стану валків прокатних станів, виявляючи незбалансованість або знос.
- Моніторинг тиску в трубопроводах або резервуарах. Наприклад, у нафтохімічній промисловості сенсори тиску контролюють рівень тиску в резервуарах для зберігання газу, запобігаючи аварійним ситуаціям.
- Сенсори положення відстежують рух і позицію механізмів. Наприклад, у робототехніці сенсори положення використовуються для точного контролю

положення маніпуляторів і забезпечення їхньої координації в процесі збирання деталей [7,13].

Застосування сенсорів:

- Постійний контроль параметрів для запобігання поломок. Наприклад, у виробництві електроенергії сенсори температури і вібрації встановлюються на турбінах для постійного моніторингу їх стану, запобігаючи аваріям.
- Раннє попередження про потенційні проблеми, що дозволяє зменшити час простоїв. Наприклад, у транспортній інфраструктурі сенсори тиску в шинах вантажних автомобілів попереджають про зниження тиску, що може призвести до аварійних ситуацій.
- Дані від сенсорів допомагають налаштувати роботу обладнання для досягнення максимального ефекту. Наприклад, у сільському господарстві сенсори вологості ґрунту допомагають оптимізувати полив, забезпечуючи ефективне використання водних ресурсів і підвищення врожайності [7,19].

Big Data та аналітика є ключовими складовими сучасних промислових процесів, дозволяючи збирати, обробляти та аналізувати величезні обсяги даних з різних джерел. Завдяки цьому підприємства можуть отримувати глибші інсайти, оптимізувати операції, передбачати можливі несправності та приймати обґрунтовані рішення. Використання передових алгоритмів аналітики та машинного навчання забезпечує значне підвищення ефективності та конкурентоспроможності бізнесу. Збір великих обсягів даних від сенсорів і пристроїв створює можливості для глибокого аналізу та прийняття більш обґрунтованих рішень [10, 21].

Основні можливості Big Data у промисловості:

- аналіз історичних даних для прогнозування несправностей і планування технічного обслуговування;
- виявлення "вузьких місць" у процесах і рекомендації щодо їх усунення;
- інтерактивні графіки і дашборди для легкого розуміння стану процесів та обладнання.

Переваги аналітики:

- швидке реагування на зміни та оптимізація процесів;
- зменшення кількості аварій і простоїв, а також раціональне використання ресурсів;
- більш точні дані дозволяють керівникам приймати обґрунтовані рішення [15,24].

Взаємодія Інтернету речей (IoT) з іншими технологіями формує нові можливості для автоматизації та оптимізації промислових процесів. IoT не лише забезпечує збір і передачу даних з різних сенсорів і пристроїв, але й інтегрується з такими технологіями, як штучний інтелект, хмарні обчислення, блокчейн і великі дані. Така синергія дозволяє створювати інтелектуальні системи, що здатні самостійно аналізувати інформацію, навчатися на основі отриманих даних і приймати рішення в режимі реального часу, що значно підвищує ефективність виробництв та знижує витрати. IoT у промисловості працює в тісній взаємодії з іншими сучасними технологіями, такими як штучний інтелект (AI), машинне навчання (ML) і хмарні обчислення.

Штучний інтелект (AI), машинне навчання (ML) використовуються для прогнозного аналізу, оптимізації процесів і автоматизації рішень [6,30].

Впровадження аналітичних інструментів та алгоритмів штучного інтелекту дозволяє отримувати цінні інсайти з даних, зібраних IoT-системами, що допомагає оптимізувати процеси, виявляти аномалії, прогнозувати поломки обладнання та приймати обґрунтовані рішення.

Хмарні обчислення забезпечують зберігання, обробку та аналіз великих обсягів даних, отриманих від IoT-пристроїв. Використання хмарних платформ дозволяє швидко масштабувати системи, підвищувати їх надійність і безпеку, а також забезпечувати доступ до даних з будь-якої точки світу [9,24].

Кібербезпека є критично важливою складовою сучасних систем Інтернету речей (IoT) та автоматизації промислових процесів. З розвитком цифрових технологій та збільшенням кількості підключених пристроїв, зростають і ризики, пов'язані з кіберзагрозами.

Для забезпечення зв'язку між різними елементами IoT-систем використовуються різні мережеві технології, такі як Wi-Fi, Bluetooth, Zigbee, LoRa та 5G. Дані технології дозволяють передавати дані від сенсорів до хмарних платформ для подальшої обробки та аналізу [12,27].

Аналіз існуючих рішень у сфері автоматизації промислових процесів та IoT показує, що такі технології мають величезний потенціал для підвищення ефективності та продуктивності виробничих процесів. Використання IoT у поєднанні з мікропроцесорними системами дозволяє створювати інтелектуальні системи, які можуть збирати, аналізувати та обробляти дані в режимі реального часу, забезпечуючи оперативне прийняття рішень й оптимізацію процесів. Успішна інтеграція цих технологій є важливою умовою для забезпечення конкурентоспроможності та сталого розвитку сучасних підприємств [5,28].

1.2 Мікропроцесорні системи у промисловій автоматизації

Мікропроцесорні системи є основою багатьох сучасних технологій, включаючи автоматизацію промислових процесів. Вони складаються з мікропроцесора, який виконує обчислення та управління, а також інших компонентів, таких як пам’ять, порти вводу-виводу та периферійні пристрої.

Основні компоненти мікропроцесорної системи:

Мікропроцесор − центральний обчислювальний елемент, який виконує інструкції програми. Включає арифметико-логічний пристрій (АЛП), регістри та блок управління.

Оперативна пам’ять (ОЗП) для тимчасового зберігання даних та інструкцій. Постійна пам’ять (ПЗП) для зберігання програмного забезпечення та даних, які не змінюються.

Порти вводу-виводу − інтерфейси для підключення зовнішніх пристроїв, таких як сенсори, актуатори та інші периферійні пристрої.

Периферійні пристрої − таймери, контролери переривань, аналогово-цифрові перетворювачі (АЦП) та інші компоненти, які розширюють функціональність системи [5].

Функції мікропроцесорних систем у промисловій автоматизації:

Виконання програм для управління виробничими процесами. Моніторинг параметрів процесу та прийняття рішень у реальному часі.

Збір даних з сенсорів та їх обробка для аналізу та прийняття рішень. Передача даних на IoT-платформи для подальшого аналізу та зберігання [4].

Аналіз даних для прогнозування можливих збоїв та поломок обладнання. Зниження часу простою та витрат на ремонт [4].

Використання протоколів зв'язку, таких як MQTT та CoAP, для забезпечення надійної передачі даних між мікропроцесорними системами та IoT-платформами. Віддалений контроль та управління виробничими процесами через IoT [4].

Використання мікропроцесорних систем у промисловості:

Автомобільна промисловість − управління роботами та маніпуляторами для складання автомобілів, моніторинг стану обладнання [4].

Електроніка − автоматизація виробництва електронних компонентів, контроль процесів пайки та монтажу [4].

Харчова промисловість − автоматизація процесів пакування та обробки продуктів, швидка зміна налаштувань для різних типів продукції [4].

Інтеграція мікропроцесорних систем з IoT для автоматизації промислових процесів дозволяє значно підвищити ефективність виробництва та забезпечити конкурентоспроможність підприємств на ринку.

1.3 Інтернет речей (IoT) та його застосування в промисловості

Інтернет речей (IoT) революціонізує промисловість, дозволяючи підключати різноманітні пристрої, датчики та обладнання до мережі для збору та обміну даними. Це відкриває нові можливості для оптимізації виробничих процесів, підвищення ефективності та зниження витрат. Розглянемо основні IoT-протоколи,

сенсорні мережі та платформи, які використовуються для автоматизації промислових процесів.

IoT-протоколи визначають правила комунікації між пристроями в мережі. Вибір протоколу залежить від таких факторів, як дальність передачі даних, пропускна здатність, енергоспоживання та вимоги до безпеки. Серед найпопулярніших IoT-протоколів можна виділити:

MQTT (Message Queuing Telemetry Transport) − легкий протокол обміну повідомленнями, оптимізований для передачі даних з сенсорів та інших пристроїв з обмеженими ресурсами. Часто використовується в системах з обмеженими ресурсами. Використовується для передачі даних у реальному часі між пристроями та серверами [7].

CoAP (Constrained Application Protocol) − протокол, розроблений спеціально для IoT-пристроїв, який забезпечує низьку затримку та ефективну передачу даних в умовах обмеженої пропускної здатності. Використовується для передачі даних між сенсорами та IoT-платформами [7].

HTTP (HyperText Transfer Protocol)/ HTTPS (HyperText Transfer Protocol Secure) − стандартний протокол для передачі даних в Інтернеті, який також використовується в IoT для доступу до вебсервісів та хмарних платформ [7].

AMQP (Advanced Message Queuing Protocol) − протокол для орієнтованої на повідомлення передачі даних, який забезпечує високу надійність і масштабованість. Використовується для передачі даних між сенсорами, контролерами та центральними системами управління [8]. Забезпечує надійну передачу повідомлень між різними бізнес-додатками та системами [9].

LoRaWAN (Long Range Wide Area Network) − протокол для довготривалої передачі даних на великі відстані з низьким енергоспоживанням, що робить його ідеальним для IoT-пристроїв, які працюють від системи живлення. Використовується для підключення сенсорів у віддалених або важкодоступних місцях [7].

Сенсорні мережі складаються з великої кількості датчиків, які збирають дані про фізичні або екологічні умови. Дані передаються через мережу для подальшої

обробки та аналізу. Сенсорні мережі широко використовуються в промисловості для моніторингу виробничих процесів, контролю якості продукції та виявлення аварійних ситуацій.

Типи датчиків:

- температурні;
- вологості;
- тиску;
- вібрації;
- газові;
- зорові;
- акустичні [5,23].

Архітектура сенсорних мереж:

- вузли датчиків збирають дані та передають їх до базової станції;
- базова станція збирає дані від вузлів датчиків та передає їх на сервер;
- сервер зберігає і обробляє дані, а також приймає рішення на основі отриманої інформації.

IoT-платформи надають набір інструментів і сервісів для розробки, розгортання та управління IoT-рішеннями. Вони спрощують процес створення IoT-додатків, забезпечуючи такі функції, як:

Збір даних з різних джерел, включаючи датчики, пристрої та хмарні сервіси.

Обробка даних − аналіз даних, візуалізація та створення дашбордів.

Управління пристроями − віддалений контроль і управління підключеними пристроями.

Масштабування − можливість масштабування системи для підтримки великої кількості пристроїв.

Популярні IoT-платформи:

ThingSpeak − безкоштовна платформа для збору, обробки та візуалізації даних IoT.

AWS IoT Core − хмарна платформа від Amazon, яка забезпечує інструменти для збирання, обробки та аналізу даних з IoT-пристроїв. Підтримує різні протоколи та інтеграцію з іншими сервісами AWS.

Microsoft Azure IoT Hub − хмарна платформа від Microsoft для безпечного і масштабованого підключення IoT-пристроїв. Платформа від Microsoft для створення, управління та моніторингу IoT-рішень. Підтримує широкий спектр пристроїв та протоколів, а також інтеграцію з іншими сервісами Azure [10].

Google Cloud IoT Core − хмарна платформа від Google для створення IoT-рішень. Платформа від Google для збирання, обробки та аналізу даних з IoT-пристроїв. Забезпечує інструменти для машинного навчання та аналітики [10].

IBM Watson IoT − платформа від IBM для створення інтелектуальних IoT-рішень. Підтримує аналітику в реальному часі та машинне навчання [10].

Застосування IoT в промисловості використовується в прогнозне обслуговування, тобто запобігання поломкам обладнання шляхом моніторингу його стану та прогнозування часу до відмови.

Оптимізація енергоспоживання, а саме зменшення витрат на енергію за рахунок автоматичного регулювання споживання в залежності від попиту.

Контроль якості продукції, полягає у забезпеченні відповідності продукції встановленим стандартам якості.

Логістика та управління ланцюгами поставок − відстеження руху матеріалів та готової продукції, оптимізація маршрутів доставки.

Безпека праці, тобто моніторинг умов праці та виявлення потенційних небезпек [11,26].

1.4 Інтеграція IoT з мікропроцесорними системами

Інтеграція Інтернету речей (IoT) з мікропроцесорними системами відкриває нові можливості для автоматизації промислових процесів. Завдяки цій інтеграції, промислові підприємства можуть збирати великі обсяги даних з різних датчиків,

аналізувати їх в режимі реального часу та приймати обґрунтовані рішення для оптимізації виробництва.

Ардуїно платформа є однією з найпопулярніших для розробки прототипів IoT-пристроїв завдяки своїй доступності, простоті використання та великій спільноті розробників (рис.1.1). Вона підтримує широкий спектр сенсорів та модулів, що дозволяє швидко створювати та тестувати IoT-рішення [11].

Рис.1.1 Плата Arduino

Особливості:

- проста у використанні платформа з відкритим кодом;
- велика кількість доступних датчиків та модулів;
- спільнота розробників, яка постійно розширює функціональність платформи;
- низька вартість [13,25].

Застосування в IoT:

- збір даних з датчиків (температура, вологість, світло тощо);
- управління актуаторами (моторами, реле, світлодіодами);
- прості логічні операції та прийняття рішень;
- створення прототипів IoT-пристроїв [2,29].

Переваги:

- легкість навчання та використання;
- велика кількість прикладів та посібників;
- низька вартість входу в розробку.

Обмеження:

- обмежена обчислювальна потужність;
- не підходить для складних обчислень.

Raspberry Pi − це міні-комп’ютер також широко використовується для IoT-проєктів (рис.1.2). Він має більшу обчислювальну потужність порівняно з Ардуїно, що дозволяє виконувати більш складні завдання, такі як обробка зображень та машинне навчання [12].

Рис.1.2 Мікропроцесорна система Raspberry Pi

Особливості:

- повноцінний одноплатний комп'ютер з операційною системою Linux;
- велика кількість портів вводу-виводу;
- підтримка різних мережевих інтерфейсів (Wi-Fi, Ethernet);
- можливість підключення дисплеїв та периферійних пристроїв.

Застосування в IoT:

- створення складних IoT-систем з великою кількістю датчиків та актуаторів;
- обробка великих обсягів даних;
- виконання складних алгоритмів машинного навчання;
- розгортання вебсерверів та хмарних сервісів.

Переваги:

- висока обчислювальна потужність;
- гнучкість конфігурації;

– велика кількість доступних програмних бібліотек.

Обмеження:

– вища вартість, ніж у Arduino;

– більш складний процес налаштування та програмування [12].

Для наочного порівняння характеристик, можливостей та сфер застосування платформ Arduino і Raspberry Pi, основні параметри цих мікропроцесорних систем узагальнено в табл. 1.2.

Таблиця 1.2

Порівняння Arduino та Raspberry Pi

Характеристика	**Arduino**	**Raspberry Pi**
Цільова аудиторія	Початківці, хобі, прототипування	Розробники, інженери, ентузіасти
Обчислювальна потужність	Низька	Висока
Вартість	Низька	Вища
Складність програмування	Проста	Більш складна
Операційна система	Немає	Linux
Застосування	Прості IoT-проєкти, автоматизація домашніх систем	Складні IoT-проєкти, промислова автоматизація, робототехніка

Вибір між Arduino та Raspberry Pi залежить від конкретних вимог проєкту.

Arduino підходить для простих IoT-проєктів, які не вимагають великої обчислювальної потужності.

Raspberry Piпідходить для складних IoT-проєктів, які потребують високої обчислювальної потужності та гнучкою [8,26].

Спеціалізовані мікроконтролери:

Texas Instruments пропонує широкий асортимент мікроконтролерів, оптимізованих для IoT-застосувань, таких як серія MSP430, яка відома своїм низьким енергоспоживанням [12].

Характеристики Texas Instruments серії MSP430 (рис.1.3):

- Найвідоміша особливість MSP430. Завдяки цьому вони ідеально підходять для автономних пристроїв, які живляться від акумуляторів або енергії навколишнього середовища.
- MSP430 мають широкий спектр вбудованих периферійних модулів, таких як АЦП, таймери, комунікаційні інтерфейси (UART, I2C, SPI), що робить їх універсальними для різних IoT-застосувань.
- MSP430 мають відносно просту архітектуру, що полегшує їхнє програмування, особливо для новачків [4,14].

Рис.1.3 Мікроконтролер Texas Instruments серії MSP430

Типові застосування:

- сенсорні вузли в системах автоматизації будинку;
- пристрої для моніторингу довкілля;
- медичні пристрої;
- безпекові системи.

STMicroelectronics − виробляє мікроконтролери серії STM32, які підтримують різні протоколи зв’язку та мають високу продуктивність, що робить їх ідеальними для IoT-додатків [12].

Рис.1.4 Мікроконтролер STMicroelectronics серії STM32

Характеристики STMicroelectronics серії STM32 (рис.1.4):

- Мікроконтролери STM32 базуються на ядрах Cortex-M, що забезпечує високу обчислювальну потужність, необхідну для складних IoT-додатків.
- Серія STM32 пропонує широкий вибір моделей з різними характеристиками, що дозволяє підібрати оптимальний мікроконтролер для конкретного завдання.
- STM32 підтримують різноманітні протоколи зв'язку, включаючи Ethernet, USB, CAN, що робить їх придатними для інтеграції в складні промислові системи.
- Багато моделей STM32 мають вбудовані модулі безпеки, графічні акселератори, що робить їх придатними для більш вимогливих застосувань [10, 21].

Типові застосування:

- промислова автоматизація;
- робототехніка;
- медичне обладнання;
- системи безпеки.

Вибір між MSP430 та STM32 залежить від конкретних вимог вашого проєкту:

Якщо пріоритетом є низьке енергоспоживання та простота використання, то MSP430 можуть бути кращим вибором [6,24].

Якщо потрібна висока продуктивність, широкий спектр периферії та підтримка складних протоколів, то STM32 є більш доцільним варіантом.

Якщо бюджет обмежений, то MSP430 можуть бути більш економічним рішенням.

Основні характеристики, переваги та сфери застосування мікроконтролерів Texas Instruments MSP430 та STMicroelectronics STM32 для IoT-завдань узагальнено у табл. 1.3 для зручності порівняння.

Таблиця 1.3

Порівняння мікроконтролерів Texas Instruments MSP430 та STMicroelectronics STM32 для IoT-застосувань

Характеристика	**MSP430**	**STM32**
Архітектура	RISC, спеціалізована для низького енергоспоживання	ARM Cortex-M, висока продуктивність
Енергоспоживання	Дуже низьке, ідеально для батарейного живлення	Середнє, баланс між продуктивністю та енергоспоживанням
Продуктивність	Нижча, але достатня для багатьох IoT-завдань	Висока, підходить для складних обчислень і багатозадачності
Периферія	Базовий набір периферії, достатній для більшості IoT-проєктів	Широкий спектр периферії, включаючи високошвидкісні інтерфейси, DSP, FPU
Ціна	Зазвичай нижча	Зазвичай вища
Розмір	Компактні, ідеальні для невеликих пристроїв	Можуть бути більшими, залежно від моделі
Розробницькі інструменти	Прості та інтуїтивно зрозумілі	Більш складні, але з більшими можливостями

Продовження таблиці 1.3

Спільнота розробників	Велика, але менша за STM32	Дуже велика, з великою кількістю бібліотек і прикладів
Типові застосування	Сенсорні вузли, моніторинг, пристрої з батарейним живленням	Промислова автоматизація, робототехніка, складні IoT-системи

Крім Texas Instruments та STMicroelectronics, існують й інші виробники, які пропонують цікаві рішення для IoT:

NXP Semiconductors відомі своїми мікроконтролерами LPC, які широко використовуються в промисловій автоматизації.

Microchip Technology пропонує широкий асортимент мікроконтролерів PIC і AVR, які відрізняються високою надійністю і довгим терміном служби.

Nordic Semiconductor спеціалізується на мікроконтролерах для бездротових застосувань, таких як Bluetooth Low Energy [6,14].

Системи на кристалі (SoC) інтегрують процесор, пам’ять та периферійні пристрої на одному чіпі, що знижує витрати та підвищує надійність. Вони забезпечують високу продуктивність та енергоефективність, що є критично важливим для IoT-пристроїв.

System-on-a-Chip (SoC) − це інтегральна схема, яка об'єднує на одному кристалі всі необхідні компоненти для функціонування електронного пристрою, а саме можуть бути:

Процесор − центральний обчислювальний блок.

Пам'ять − оперативна пам'ять (RAM), постійна пам'ять (ROM) для зберігання програмного забезпечення.

Периферійні пристрої − АЦП, ЦАП, таймери, інтерфейси (UART, I2C, SPI, USB), мережеві контролери (Wi-Fi, Bluetooth, Ethernet) та багато інших [15,25].

Переваги використання SoC в IoT:

– завдяки високому ступеню інтеграції SoC дозволяють створювати мініатюрні пристрої;

– оптимізовані процеси та низьке енергоспоживання периферії сприяють збільшенню часу автономної роботи пристроїв;
– сучасні SoC забезпечують достатню обчислювальну потужність для виконання складних алгоритмів, таких як машинне навчання;
– масове виробництво знижує вартість SoC, що робить їх доступними для широкого кола розробників;
– наявність готових розробок та бібліотек спрощує процес створення IoT-пристроїв [3,29].

Існує багато популярних SoC, які використовуються в IoT-проєктах:

Nordic nRF52 − спеціалізовані SoC для бездротових застосувань, що підтримують Bluetooth Low Energy, 2.4 GHz ISM та інші стандарти.

STM32 − широка лінійка мікроконтролерів від STMicroelectronics з високою продуктивністю та різноманітними периферійними модулями.

RP2040 − новий мікроконтролер від Raspberry Pi Foundation, який обіцяє високу продуктивність за доступною ціною.

Qualcomm Snapdragon SoC, що використовуються в смартфонах, також знаходять застосування в IoT-пристроях, які вимагають високої обчислювальної потужності та можливостей обробки мультимедіа [8,16].

Типові застосування SoC в IoT:

Сенсорні вузли − збір даних про температуру, вологість, світло, рух тощо.

Актуатори − управління моторами, сервоприводами, реле для автоматизації процесів.

Бездротові мережі − створення мереж датчиків для моніторингу довкілля, промислових процесів.

Розумні пристрої − створення розумних будинків, розумних міст.

Носимі пристрої − створення фітнес-трекерів, смарт-годинників.

При виборі SoC для свого IoT-проєкту необхідно враховувати такі фактори:

Якщо потрібна висока продуктивність, слід звернути увагу на SoC з потужними процесорами [3,19].

Для автономних пристроїв критичним є низьке енергоспоживання.

Наявність необхідних периферійних модулів для підключення датчиків, актуаторів та інших компонентів.

Вартість − SoC, який відповідає індивідуальному бюджету.

Для компактних пристроїв важливий фізичний розмір чіпа.

Велика спільнота розробників може значно полегшити процес розробки [12].

1.5 Фреймворків для роботи з IoT (Node-RED, AWS IoT, Azure IoT).

Інтернет речей (IoT) стає все важливішим для сучасних технологічних рішень. Для розробки та управління IoT-проєктами використовуються спеціалізовані фреймворки, які допомагають спростити інтеграцію пристроїв, збір даних і управління ними. Ось кілька популярних фреймворків для роботи з IoT:

Node-RED − це інтуїтивно зрозумілий та потужний фреймворк для візуального програмування, створений для з'єднання пристроїв, API та онлайн-сервісів. Він дозволяє легко інтегрувати різноманітні IoT-пристрої та створювати автоматизовані робочі процеси за допомогою "потокових" функцій.

Особливості:

Візуальне програмування за допомогою простого інтерфейсу користувача можна з’єднувати різні пристрої та сервіси без потреби в написанні коду.

Підтримка протоколів Node-RED підтримує популярні IoT-протоколи, такі як MQTT, HTTP, WebSockets, CoAP.

Легка інтеграція з хмарними платформами: підтримує зв'язок з платформами, такими як IBM Watson, Microsoft Azure, AWS і іншими [6,27].

Велика бібліотека готових компонентів та модулів.

Створення автоматизованих сценаріїв для розумного будинку, обробка даних з датчиків, моніторинг стану обладнання в реальному часі.

AWS IoT (Amazon Web Services Internet of Things) − це хмарна платформа від Amazon для управління IoT-пристроями та обробки даних з них. Платформа дозволяє безпечно з’єднувати пристрої, збирати і зберігати дані, а також запускати аналітику та інші сервіси.

Основні компоненти AWS IoT:

AWS IoT Core − основна служба, яка дозволяє підключати IoT-пристрої до хмари. Вона забезпечує безпечне обміну даними між пристроями та хмарою через різноманітні протоколи, такі як MQTT, HTTP і WebSockets.

Можливість підтримки мільйонів пристроїв одночасно з мінімальною затримкою[8,20].

AWS IoT Device Management − інструменти для керування та моніторингу IoT-пристроїв, що дозволяють автоматизувати процеси налаштування пристроїв, оновлення програмного забезпечення та управління їх станом.

AWS IoT Analytics − сервіс для обробки та аналізу даних, зібраних від IoT-пристроїв. Він допомагає здійснювати глибокий аналіз великих обсягів даних, забезпечуючи виведення інсайтів для прийняття рішень.

AWS IoT Events − система для виявлення подій і аномалій в даних, що надходять від IoT-пристроїв, що дозволяє автоматизувати реакції на певні зміни в стані пристроїв.

AWS IoT Greengrass − дозволяє IoT-пристроям виконувати обчислення, зберігання та обробку даних локально, а також підтримує функції машинного навчання та управління пристроями навіть без постійного з'єднання з хмарою.

AWS IoT Device Defender − інструмент для забезпечення безпеки пристроїв, який допомагає моніторити їх поведінку, виявляти аномалії та забезпечувати дотримання політик безпеки.

Azure IoT − це набір хмарних сервісів від компанії Microsoft, який надає можливості для підключення, управління, обробки та аналізу даних з IoT-пристроїв через хмару. Платформа дозволяє створювати масштабовані, надійні та безпечні рішення для Інтернету речей (IoT), інтегруючи пристрої з іншими сервісами Microsoft, такими як аналітика, машинне навчання, штучний інтелект та бізнес-інтелект [7,29].

Основні компоненти Azure IoT:

Azure IoT Hub − це основна служба для підключення і керування IoT-пристроями. Вона забезпечує двосторонню комунікацію між пристроями та

хмарою, підтримуючи такі протоколи, як MQTT, AMQP та HTTPS. Забезпечує можливості для аутентифікації пристроїв, шифрування даних, моніторингу та керування пристроями в реальному часі.

Azure IoT Central − платформа для створення та керування IoT-додатками без необхідності написання великої кількості коду. Вона надає готові шаблони та інтерфейси для підключення пристроїв, моніторингу та керування даними. Спрощує розгортання та моніторинг IoT-систем, дозволяючи зосередитись на бізнес-логіці замість інфраструктури.

Azure Digital Twins − цей сервіс дозволяє створювати цифрові моделі реальних об'єктів та середовищ для візуалізації, моніторингу і взаємодії з фізичними системами. Він допомагає симулювати різні сценарії, а також прогнозувати поведінку систем на основі даних. Використовується для створення моделей складних фізичних середовищ, таких як розумні міста або промислові об'єкти [6,20].

Azure IoT Edge − це рішення дозволяє здійснювати обробку даних на місці, тобто на самому пристрої, без необхідності постійно передавати дані в хмару, що дозволяє знижувати затримки та зменшувати навантаження на мережу. Підтримує виконання контейнерних додатків, машинного навчання та аналітики прямо на пристрої, яке дозволяє працювати навіть при нестабільному з'єднанні з Інтернетом.

Azure Stream Analytics − сервіс для обробки потокових даних у реальному часі. Дозволяє аналізувати дані, які надходять від IoT-пристроїв, і приймати рішення на основі аналізу цих даних. Вбудовані можливості для інтеграції з іншими сервісами Azure, такими як Azure Machine Learning для глибокого аналізу даних.

Azure Machine Learning − платформа для розробки, тренування та впровадження моделей машинного навчання. Вона дозволяє обробляти великі обсяги даних з IoT-пристроїв та створювати прогнози для автоматичного прийняття рішень. Інтеграція з іншими сервісами Azure для побудови комплексних аналітичних рішень, зокрема для прогнозування поломок обладнання, аналізу візуальних даних і автоматизації процесів [8,21].

Для зручності аналізу особливостей, функціональних можливостей та сфер застосування фреймворків Node-RED, AWS IoT та Azure IoT їх основні характеристики представлені у табл. 1.4.

Таблиці 1.4

Порівняння фреймворків

Характеристика	**Node-RED**	**AWS IoT**	**Azure IoT**
Тип платформи	Візуальне програмування	Хмарна платформа для IoT	Хмарна платформа для IoT
Інтерфейс	Візуальний (drag-and-drop)	Інтерфейс для керування пристроями через API	Інтерфейс для керування пристроями через API
Безпека	Підтримка базових функцій безпеки	Розширена безпека, аутентифікація та шифрування	Розширена безпека, підтримка IoT стандарту
Масштабованість	Для невеликих та середніх проєктів	Висока масштабованість для великих проєктів	Висока масштабованість для великих проєктів
Підтримка протоколів	MQTT, HTTP, WebSockets, CoAP	MQTT, HTTP, WebSockets	MQTT, HTTP, WebSockets
Інтеграція з іншими сервісами	Інтеграція з багатьма сервісами через API	Інтеграція з іншими сервісами AWS (Lambda, S3)	Інтеграція з іншими сервісами Azure (Machine Learning, Power BI)

Node-RED, AWS IoT і Azure IoT − це потужні фреймворки для розробки та управління IoT-проєктами, кожен з яких має свої переваги залежно від специфіки задачі. Node-RED підходить для проєктів, де потрібна швидка інтеграція та візуальне програмування, AWS IoT і Azure IoT − для більш масштабованих і складних проєктів з високими вимогами до безпеки та обробки даних [23,29].

У розділі було проаналізовано сучасні технології автоматизації промислових процесів та Інтернету речей (IoT). Виявлено, що поєднання мікропроцесорних систем та IoT забезпечує значну оптимізацію процесів, зменшує витрати та підвищує ефективність. Особлива увага приділена протоколам зв'язку, платформам IoT та ролі сенсорних мереж у промисловості. Даний аналіз став основою для розробки інтегрованої системи автоматизації.

РОЗДІЛ 2 РОЗРОБКА КОНЦЕПЦІЇ АВТОМАТИЗАЦІЇ

2.1 Розробка концепції автоматизації промислового електродвигуна засобами IoT

Концепція автоматизації промислового електродвигуна із використанням технологій IoT базується на інтеграції апаратних і програмних компонентів для забезпечення моніторингу, управління та аналізу роботи двигуна в реальному часі. Використання IoT дозволяє досягти високого рівня адаптивності, автоматизації та ефективності роботи обладнання, а також створює можливості для прогнозного обслуговування та зменшення простоїв [16,18].

Метою розробки концепції є створення гнучкої, масштабованої та безпечної системи автоматизації промислового електродвигуна, яка дозволяє знизити енергоспоживання, мінімізувати ризики несправностей і забезпечити високу якість виробничих процесів.

Основні принципи концепції:

1. Інтеграція IoT-пристроїв має проходити встановлення сенсорів і контролерів для збору даних про роботу електродвигуна та використання протоколів зв'язку (OPC-UA, MQTT) для передачі даних між пристроями та сервером.

2. Реалізація локального управління відбувається впровадження контролера Arduino D1 WiFi UNO R3 ESP8266 для збору даних і передачі команд частотному перетворювачу DELFA VFD004EL43A та використання локального сервера Raspberry Pi 4 Model B для попередньої обробки даних й взаємодії з користувачами через вебінтерфейс.

3. Хмарна інтеграція відбувається через зв'язок з хмарними платформами (AWS IoT Core, Azure IoT Hub) для зберігання, аналізу даних та віддаленого управління, та використання Amazon SageMaker та Azure Stream Analytics для аналітики та прогнозування.

4. Адаптивність і масштабованість легко через інтеграцію додаткових датчиків (температури, вібрації, вологості), підтримка зворотного зв'язку для динамічної зміни параметрів роботи двигуна залежно від змін умов експлуатації.

Запропонована концепція автоматизації промислового електродвигуна із використанням IoT забезпечує гнучкість, надійність та ефективність управління, відкриваючи можливості для впровадження сучасних технологій в промисловість. Її реалізація сприяє підвищенню конкурентоспроможності підприємства завдяки оптимізації виробничих процесів й зменшенню витрат на обслуговування обладнання.

2.2 Алгоритм роботи системи управління електродвигуном

Алгоритм роботи системи управління електродвигуном за допомогою IoT починається з ініціалізації системи (рис. 2.1). Спочатку запускається локальний сервер і встановлюється зв'язок з контролером та сенсорами. Важливо також перевірити доступність хмарних сервісів, таких як AWS або Azure. Потім система автоматично виявляє підключені пристрої, включаючи контролер, частотний перетворювач та датчики, і завантажує початкові параметри з бази даних, зокрема налаштування двигуна та інтерфейс користувача.

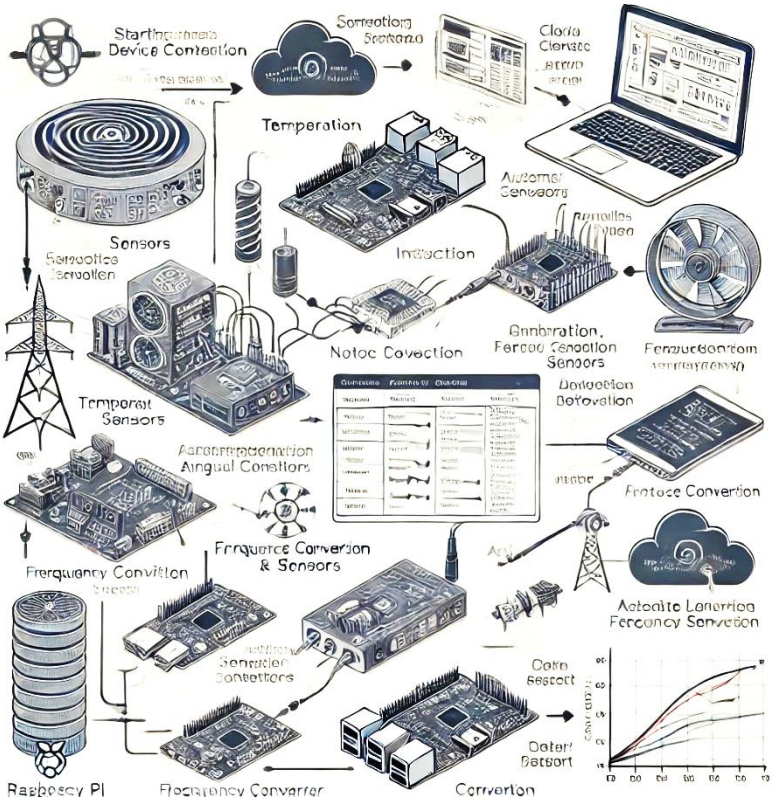


Рис.2.1 Алгоритм роботи IoT-системи управління електродвигуном

Основний цикл роботи починається зі збору даних. Сенсори зчитують показники, такі як температура, вібрація, кутове положення та напруга. Потім дані проходять через фільтрацію шумів та перевірку на коректність. Після цього починається обробка даних. Локально, на Raspberry Pi, аналізуються вхідні дані, тобто розраховується поточна ефективність двигуна, порівнюються з заданими параметрами роботи та виявляються відхилення, але, якщо необхідно, тоді визначається потреба у коригуванні [8,28]. Алгоритм регулювання швидкості приводу двигуна представлений у вигляді блок-схеми, що відображає основні етапи обробки даних та прийняття рішень (рис. 2.2).

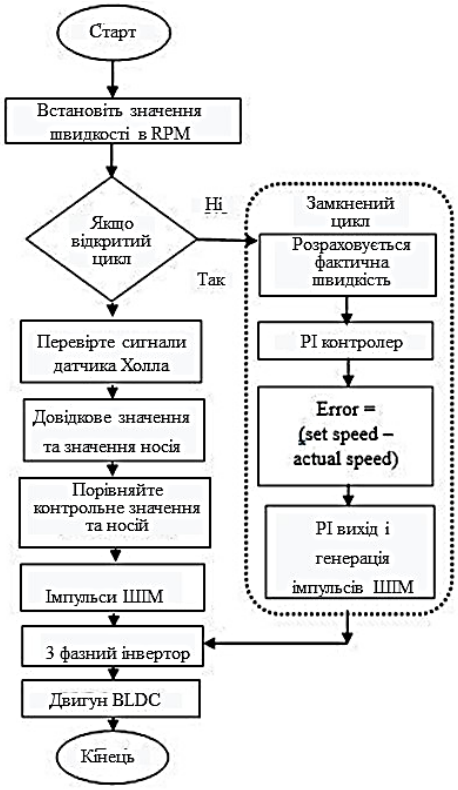


Рис.2.1 Блок-схема для регулювання швидкості приводу двигуна

На основі оброблених даних формуються команди для частотного перетворювача, які регулюють швидкість та напрямок обертання двигуна. Команди передаються через контролер Arduino. У процесі роботи всі дані записуються у локальну базу даних SQLite для короткострокового зберігання, а також надсилаються у хмару, включаючи поточні показники роботи та журнали аварійних ситуацій.

Аналіз даних у хмарі здійснюється з використанням алгоритмів машинного навчання для прогнозування поломок та генерації аналітичних звітів про ефективність та стан двигуна. Користувач може взаємодіяти із системою через вебінтерфейс, де відображаються поточні дані, а також отримувати повідомлення про критичні події через SMS або push-сповіщення. Система також забезпечує можливість дистанційного управління [7,17].

Завершення роботи системи включає відключення двигуна за запитом користувача або при виявленні критичних несправностей. Всі дані роботи зберігаються у хмару та локально. Після цього генерується звіт про стан системи на момент зупинки та перевіряється необхідність технічного обслуговування.

Даний алгоритм забезпечує послідовну роботу програмного забезпечення, гарантуючи високу ефективність і надійність управління системою автоматизації.

У розділі розроблено концепцію автоматизації промислового електродвигуна із використанням IoT. Описано етапи інтеграції апаратних та програмних компонентів, що забезпечують адаптивність, надійність і масштабованість системи. Було створено алгоритм, який дозволяє автоматично регулювати параметри двигуна та передбачати можливі несправності, що підтверджує ефективність розробленої концепції.

РОЗДІЛ 3 ПРОЄКТУВАННЯ АПАРАТНОЇ ЧАСТИНИ

3.1 Принцип роботи системи управління електродвигуном із використанням IoT

Система управління електродвигуном побудована на базі інтеграції компонентів IoT, що забезпечують автоматизацію процесів, адаптивність і можливість моніторингу в реальному часі. В її основі лежить взаємодія між частотним перетворювачем, контролером, енкодером і локальним сервером. Інтеграція системи управління електродвигуном із запропонованою архітектурою IoT з платформами AWS та Azure може значно підвищити її функціональність та ефективність. Розглянемо, як працює дана система крок за кроком [11,12].

Система налаштовується на певні параметри роботи двигуна (наприклад, швидкість обертання, напрямок руху та режими роботи). Такі налаштування вводяться через локальний сервер Raspberry Pi 4 Model B в вебінтерфейсі та мобільний додаток або ПК через підключення до Wi-Fi, що забезпечує модуль Arduino D1 WiFi UNO R3 ESP8266 ESP-12E. Задана конфігурація передається на контролер для подальшої обробки. Модуль Arduino D1 WiFi UNO R3 ESP8266 ESP-12E підключається до Azure IoT Hub, що забезпечує безпечне підключення пристроїв до хмари. А також підключається до AWS IoT Core, що дозволяє безпечно підключати пристрої до хмари. На рисунку 3.1 представлено архітектуру сервісів AWS Cloud, що забезпечує реалізацію функцій аналітики та прогнозування у системі автоматизації.

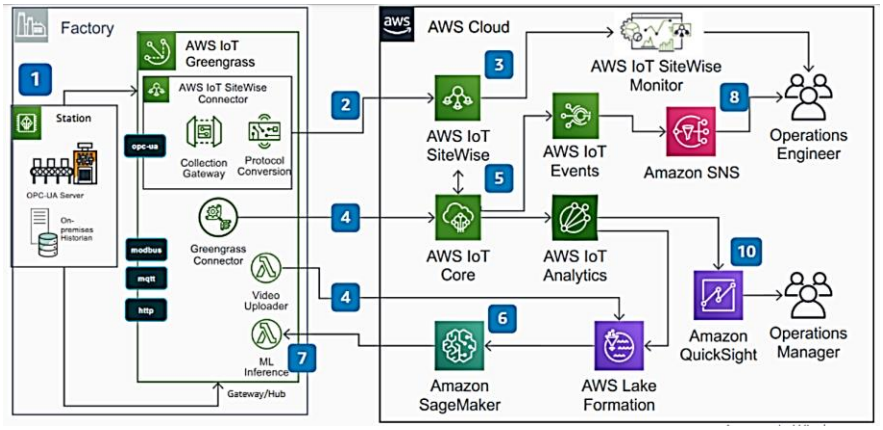


Рис.3.1 Архітектура сервісів AWS Cloud для рішення аналітики та прогнозування

Розглянемо архітектуру сервісів AWS Cloud для рішення аналітики та прогнозування (рис. 3.1):

1 - промислове обладнання з IoT, яке генерує дані;

2 - канал збору даних із заводських машин через OPC-UA, що забезпечує комунікацію між локальним AWS IoT SiteWise Connector та хмарним AWS IoT SiteWise;

3 - сервіс AWS IoT SiteWise, який відповідає за масштабування та зберігання мережевих даних;

4 - основний канал зв’язку локального сервера AWS IoT Greengrass із глобальною мережею Amazon AWS IoT Core для обміну повідомленнями;

5 - сервіс інтеграції даних між AWS IoT Core та AWS IoT Event, який використовується для реагування на події IoT та їх аналітики;

6 - сервіс прогнозування якості із використанням моделей машинного навчання через Amazon SageMaker, який аналізує зображення, збережені в AWS Lake Formation;

7 - локальна платформа машинного навчання на основі шлюзу AWS IoT Greengrass Edge;

8 - канал сповіщень через Amazon Simple Notification Service (SNS);

9 - вебпортал для візуалізації даних у режимі реального часу;

10 - аналітичний портал, який надає статистику оброблених даних [9,29].

Контролер Arduino D1 ініціює запуск електродвигуна, передаючи сигнал частотному перетворювачу DELFA VFD004EL43A, та контролює поточні параметри роботи двигуна, отримуючи дані від енкодера. А енкодер Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R постійно вимірює швидкість обертання валу двигуна та його кутове положення й передає ці дані в цифровому форматі на Arduino для аналізу. Дані з енкодера Opkon та частотного перетворювача DELFA передаються на Azure IoT Hub, а також дані з енкодера Opkon та частотного перетворювача DELFA передаються на AWS IoT Core.

На основі отриманих від енкодера даних контролер порівнює поточні показники із заданими параметрами. У разі виявлення відхилень Arduino D1 генерує коригуючі команди. Дані команди передаються на частотний перетворювач DELFA, який змінює частоту змінного струму, що подається на двигун. Регулює напругу для забезпечення стабільної роботи двигуна. Даний процес відбувається в реальному часі, що дозволяє оперативно адаптувати роботу системи до змінних умов (наприклад, підвищеного навантаження).

Зібрані дані про роботу двигуна передаються на локальний сервер Raspberry Pi 4 Model B. Azure IoT Hub збирає дані з пристроїв і передає їх на Azure Stream Analytics для обробки в реальному часі. Його завдання зберігати дані, де уся інформація записується для подальшого аналізу та звітності. Використання Azure Functions для обробки даних та прийняття рішень на основі отриманих даних. Тоді відбувається візуалізація параметрів, а саме дані відображаються у вигляді графіків, таблиць чи індикаторів у вебінтерфейсі. Дані зберігаються в Azure Blob Storage або Azure Cosmos DB для подальшого аналізу та зберігання історичних даних. В результаті чого сервер аналізує отриману інформацію для виявлення ознак несправностей або зниження ефективності. Використання Azure Monitor для моніторингу стану системи та отримання сповіщень про можливі проблеми. Управління пристроями через Azure IoT Device Management. А також AWS IoT Core збирає дані з пристроїв і передає їх на AWS IoT Analytics для подальшого аналізу. Використання AWS Lambda для обробки даних у реальному часі та прийняття рішень на основі отриманих даних. Дані зберігаються в Amazon S3 або

Amazon DynamoDB для подальшого аналізу та зберігання історичних даних. Використання AWS CloudWatch для моніторингу стану системи та отримання сповіщень про можливі проблеми. Управління пристроями через AWS IoT Device Management [13,15].

На рисунку 3.2 зображена архітектура системи моніторингу ресурсів в AWS Cloud, яка забезпечує збір, обробку та аналіз даних для ефективного управління ресурсами.

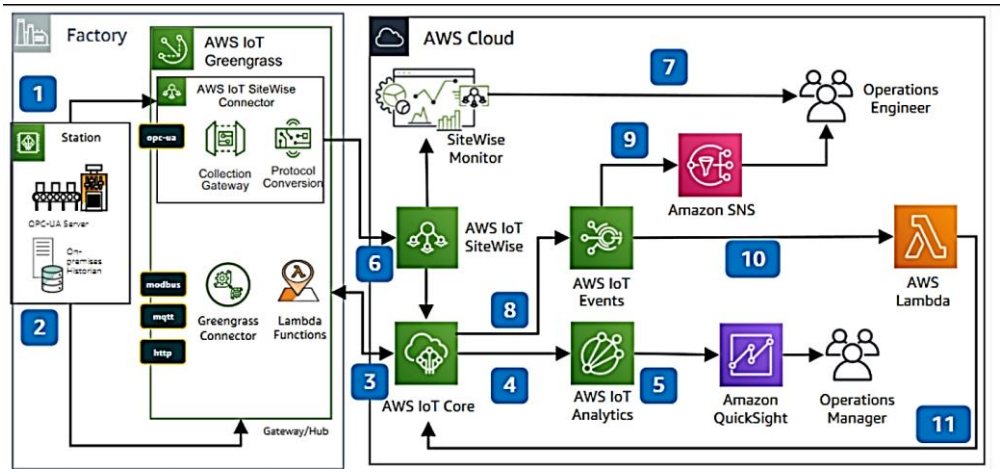


Рис. 3.2 Архітектура системи AWS Cloud моніторингу ресурсів

Розберемо архітектуру системи AWS Cloud моніторингу ресурсів (рис. 3.2):

1 - промислове обладнання з IoT, яке є джерелом даних;

2 - сервіс локальної обробки та аналізу даних безпосередньо на підприємстві;

3 - канал мережевого ядра AWS IoT Core, який ініціює події та пересилає їх до служб AWS IoT Events та AWS IoT Analytics;

4 - канал передачі даних до мережевих служб для швидкого аналізу;

5 - сервіс обробки даних, що відповідає за масштабування, статистичний аналіз та аналітику;

6 - сервіс моделей, який зберігає дані та відслідковує їх зміни;

7 - вебпортал AWS IoT SiteWise Monitor для візуалізації даних у реальному часі;

8 - канал служби повідомлень, який забезпечує відслідковування подій;

9 - служба публікації та розсилки повідомлень, що діє на основі подій;

10 - сервіс мережевих обчислень, що забезпечує складні розрахунки;

11 - канал взаємодії між AWS IoT Core та AWS Lambda, що забезпечує інтеграцію з функціями Lambda [14,18].

Система підтримує зворотний зв'язок, який дозволяє вносити корективи в режим роботи двигуна, тобто якщо енкодер фіксує відхилення (наприклад, зменшення швидкості через підвищене навантаження), система автоматично коригує параметри роботи частотного перетворювача. А якщо сервер виявляє ознаки критичних умов (перевищення температури, перевантаження), він надсилає сигнал для зупинки двигуна або переходу в енергоощадний режим [11,19].

На рисунку 3.3 представлена архітектура системи AWS Cloud для аналітики технічного обслуговування, яка включає інструменти для збору, обробки та аналізу даних з метою оптимізації процесів обслуговування та прогнозування несправностей.

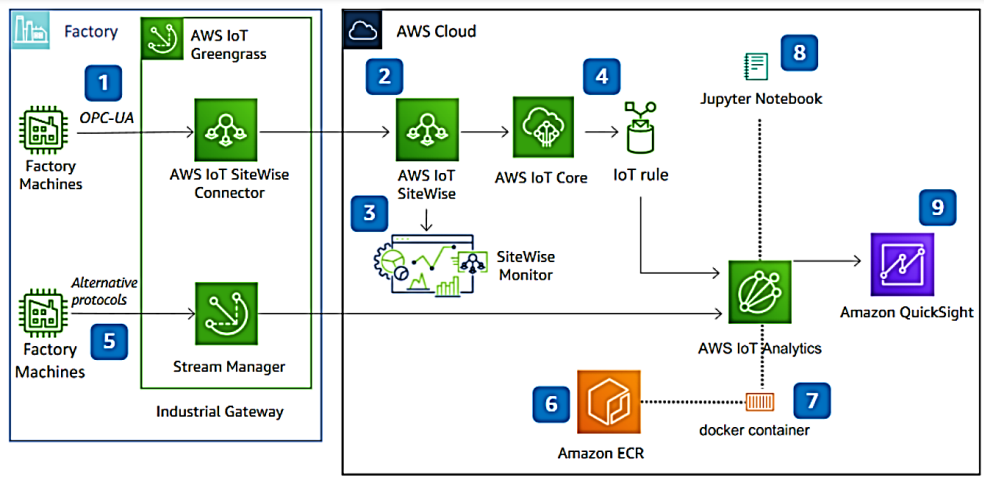


Рис. 3.3 Архітектура системи AWS Cloud аналітики технічного обслуговування

Розберемо архітектуру системи AWS Cloud аналітики технічного обслуговування (рис. 3.3):

1 - канал зв’язку для підключення та збору даних із заводського обладнання за допомогою протоколу OPC-UA;
2 - сервіс моделей, який відповідає за збереження даних та відстеження їх змін;
3 - вебпортал для візуалізації даних із заводу та показників ефективності виробництва в реальному часі;
4 - сервіс резервного копіювання та передачі повідомлень для аналітичної обробки;
5 - менеджер потоків, що забезпечує паралельну та швидку обробку даних;
6 - менеджер стану системи, який керує обробленими даними;
7 - контейнери Docker, що використовуються для ізольованого виконання процесів;
8 - та 9 - додаткові інструменти, призначені для інтерактивної взаємодії з користувачами [14,24].

Завдяки модулю Wi-Fi ESP8266 користувач має можливість підключатися до системи через мобільний пристрій чи ПК та переглядати поточні параметри роботи двигуна. Змінювати налаштування в реальному часі (наприклад, задавати нові швидкості обертання). Крім того, сервер Raspberry Pi може передавати дані на хмарні платформи для довгострокового зберігання або віддаленого аналізу. Система легко інтегрується з додатковими сенсорами й пристроями, такими як датчики температури для контролю перегріву двигуна, і датчики вібрації для виявлення механічних несправностей. Системи прогнозного обслуговування для запобігання поломкам завдяки аналізу зібраних даних [21,30].

3.2 Апаратна складова системи автоматизації промислового електродвигуна засобами з IoT

Розглянемо більш детально систему управління електродвигуном із запропонованою архітектурою реалізації IoT, яка містить: частотний перетворювач DELFA VFD004EL43A, Модуль arduino D1 WiFi UNO R3 ESP8266

ESP-12E, енкодер Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R, Локальний сервер Raspberry Pi 4 Model B (рис. 3.4). Дана система керування може бути реалізована в роботизованій системі [7, 8] з незначними змінами та в промисловості для будівельних кранів, машин для дроблення матеріалів [9] та ущільнення.

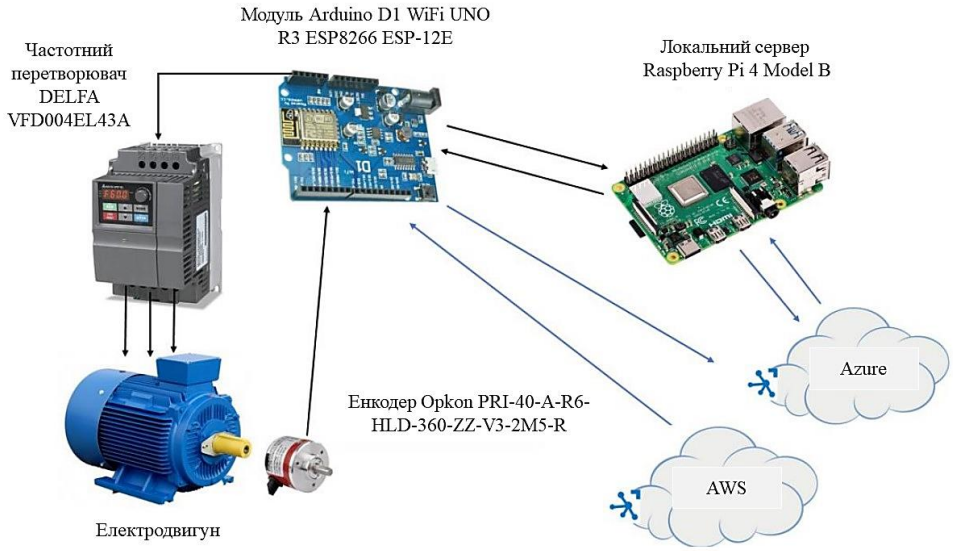


Рис. 3.4 Запропонована система автоматизації промислового електродвигуна засобами з IoT

Складові системи управління електродвигуном**:**

1. Частотний перетворювач DELFA VFD004EL43A використовується для регулювання швидкості обертання електродвигуна шляхом зміни частоти живлення (рис. 3.5).

Основні характеристики DELFAVFD004EL43A:

1) Частотний перетворювач для однофазних і трифазних електродвигунів змінного струму, підходить для двигунів на змінному струмі (AC).

2) Електричні характеристики:

- вхідна напруга:

  1-фазне підключення: 220 V (±15%);

  3-фазне підключення: 380 V (±15%);

- регульована змінна напруга від 0 до номінальної вхідної напруги;
- регулювання частоти в діапазоні 0 − 400 Гц;

– пульсширина модуляції (PWM) керування для точного регулювання швидкості та моменту обертання;
– підтримка двигунів на змінному струмі (AC) з функцією регулювання швидкості та потужності [7,28].

Рис. 3.5 Частотний перетворювач DELFA VFD004EL43A

3) Максимальна потужність для однофазних двигунів 0,4 кВт (в залежності від версії та модифікації), для трифазних двигунів може підтримувати потужність до 4 кВт. Максимальний вихідний струм 5 А (для стандартної моделі).

4) Температурний діапазон зазвичай від -10°C до +50°C, що робить його підходящим для використання в різних умовах, включаючи промислові застосування.

5) Стандартне керування через потенціометр або цифрове управління через зовнішні пульти/панелі, на частотному перетворювачі може бути вбудований дисплей для моніторингу параметрів роботи системи, автоматичне, ручне, в залежності від підключених зовнішніх пристроїв.

6) Повітряне охолодження для забезпечення стабільної роботи. Захист від короткого замикання, перегріву, перевантаження по струму, перенапруги та пониженої напруги. Вбудовані механізми безпеки, які автоматично зупиняють роботу при перевантаженнях або неполадках.

7) Деякі моделі можуть підтримувати модулі зв'язку, такі як RS485 (Modbus), що дозволяє здійснювати віддалене керування через мережу. Існують варіанти для підключення зовнішніх датчиків і контролерів, таких як потенціометри, сенсори температури, зворотний зв'язок з енкодерами для точного контролю швидкості [9,11].

8) Фізичні характеристики:

- розміри приблизно 140 × 95 × 155 мм (в залежності від моделі);
- вага пристрою варіюється від 1,5 до 3 кг залежно від потужності та модифікації.

9) Частотні перетворювачі типу DELFA VFD004EL43A широко використовуються в промисловості для автоматизації процесів і регулювання швидкості обертання електродвигунів. Дані перетворювачі часто використовуються в будівельних кранах, насосах, вентиляторах, конвеєрних системах, а також в різних машинах для дроблення та ущільнення матеріалів [16,19].

Переваги використання:

- можливість точного контролю швидкості обертання електродвигунів у широкому діапазоні частот;
- зменшення споживаної енергії при змінних навантаженнях завдяки оптимізації роботи двигуна;
- підтримка різних типів двигунів і гнучкість у налаштуваннях [8,23].

Даний перетворювач є одним із важливих елементів для автоматизації та керування в промислових системах, де потрібна адаптація швидкості двигунів залежно від умов експлуатації.

2. Модуль arduino D1 WiFi UNO R3 ESP8266 ESP-12E виконує функції контролера та забезпечує бездротовий зв'язок (рис.3.6).

Модуль Arduino D1 WiFi (на базі ESP8266 ESP-12E) − це одна з популярних плат розробки для IoT-проєктів, що поєднує можливості ESP8266 з платформою Arduino для забезпечення бездротового зв'язку через Wi-Fi. Розглянемо основні характеристики цього модуля:

1) Процесор:

- чип ESP8266 (використовує мікроконтролер Tensilica L106);
- частота 80 MHz (можна розігнати до 160 MHz);
- кількість ядер: 1 ядро (однопроцесорний);
- розрядність 32-бітна архітектура.

2) Пам'ять:

- оперативна пам'ять (RAM) 160 KB;
- пам'ять для зберігання програм (Flash) 4 MB (може варіюватися залежно від конкретної версії плати) [9,26].

Рис. 3.6 Модуль Arduino D1 WiFi

3) Комунікаційні інтерфейси:

- підтримує Wi-Fi 802.11 b/g/n, доступ до мережі за допомогою стандартних протоколів TCP/IP, UDP, HTTP, MQTT та інших;
- 2 порти UART (TX, RX) для серійного зв'язку;
- підтримка SPI для підключення додаткових периферійних пристроїв4
- підтримка I2C для підключення сенсорів і інших пристроїв;
- 1 або більше портів для ШІМ сигналів, що дозволяє керувати яскравістю світлодіодів або швидкістю обертання моторів;
- 1 аналоговий вхід (10-бітне перетворення, до 1 В) для вимірювання аналогових сигналів.

4) Порти вводу/виводу (GPIO):

- плата має 11 цифрових I/O пінів (залежно від конкретної версії, деякі піні можуть бути функціональними для інших задач);
- всі цифрові піні підтримують як входи, так і виходи;
- 1 аналоговий пін (A0), 10-бітний аналого-цифровий перетворювач (ADC).

5) Живлення 3.3 V (важливо зазначити, що плата працює на 3.3V, тому необхідно бути обережним при підключенні пристроїв, які працюють на 5V, щоб

не пошкодити чіп), та в залежності від використання, може варіюватися від 100 мА до 300 мА при активному Wi-Fi з’єднанні.

6) Wi-Fi можливості підтримує режим точки доступу (Access Point) та режим клієнта (Station Mode). Режим Wi-Fi Direct дає можливість підключення пристроїв без необхідності в маршрутизаторі через мережеві протоколи TCP/IP, UDP, HTTP, HTTPS, MQTT, DNS, і багато інших для реалізації IoT-програм.

7) Плата легко програмується через Arduino IDE, що дозволяє використовувати бібліотеки та простий код для швидкої розробки. Бібліотеки для Wi-Fi ESP8266 має власні бібліотеки для роботи з Wi-Fi, такими як ESP8266WiFi, WiFiClient, WiFiServer. Програмування відбувається через USB порти за допомогою FTDI адаптера для серійного програмування (необхідно для Arduino D1).

8) Особливості та додаткові можливості:

- плата має компактний розмір, що зручний для інтеграції в різні проєкти;
- вбудовані інтерфейси I2C та SPI дозволяють підключати різні датчики, такі як температури, вологості, акселерометри, гіроскопи та інші;
- підтримка OTA (Over-The-Air), тобто можливість оновлення прошивки без використання проводів, через Wi-Fi з'єднання.

9) Програмні фреймворки та середовища:

- простота у використанні через підтримку популярного середовища Arduino IDE;
- інше середовище розробки, яке також підтримує програмування для ESP8266;
- чи інші середовища, які можуть бути використані для програмування ESP8266.

Завдяки можливостям Wi-Fi та простоті програмування, модуль використовується для створення розумних будинків, систем моніторингу, автоматизації. Підходить для систем з віддаленим керуванням, таких як системи керування електродвигунами, системи безпеки. Ідеально підходить для проєктів,

що потребують комунікації між пристроями через Інтернет або локальну мережу [7,28].

Таким чином, Arduino D1 WiFi ESP8266 ESP-12E є універсальним модулем, який ідеально підходить для розробки IoT-проєктів завдяки його потужним характеристикам, простоті використання та широким можливостям для зв'язку через Wi-Fi.

3. Енкодер Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R − використовується для вимірювання швидкості та положення обертання (рис.3.7).

Енкодер Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R − це тип оптичного енкодера, який використовується для вимірювання кута обертання або лінійного переміщення. Такий енкодер широко застосовується для зворотного зв'язку в системах автоматизації, таких як робототехніка, системи управління двигунами, конвеєрні лінії, а також в інших промислових застосуваннях.

Рис. 3.7 Енкодер Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R

Основні характеристики енкодера Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R:

1) Тип енкодера:
   - абсолютний оптичний енкодер;
   - принцип роботи оптичного енкодера, за допомогою фотодіодів та диска з перфорованими отворами для визначення кута обертання.
2) Фізичні характеристики:
   - діаметр корпус 40 мм (це стандартний розмір для такого типу енкодерів);

- тип підключення штекер або кабель (в залежності від моделі), що дає змогу підключати енкодер до контролера або системи керування.
- можливість кріплення на вал, з використанням фланця чи іншого механізму монтажу для фіксації енкодера в потрібному положенні.

3) Технічні параметри:

- кількість імпульсів (роздільна здатність) 360 імпульсів на один оберт (360 PPR, Pulses per Revolution) та означає, що для кожного повного оберту валу енкодер генерує 360 імпульсів;
- тип виходу;
- релєфний вихід (TTL) 5 V або 24 V, залежно від версії;
- роз'єм зазвичай використовуються типи виходу A/B (диференційний) для точного вимірювання кута обертання з високою точністю;
- точність обертання складає ±1 імпульс на оберт, що забезпечує високу точність в вимірюванні кута.

4) Електричні характеристики:

- напруга живлення зазвичай 5 V DC або 24 V DC в залежності від моделі енкодера;
- струм споживання зазвичай невеликий (близько 50-100 мА), що дозволяє підключати енкодер до більшості стандартних керуючих систем;
- частота виходу 100 кГц або вище, що дозволяє використовувати енкодер у швидких та високошвидкісних системах.

5) Механічні характеристики:

- максимальна швидкість обертання до 6000 обертів на хвилину (RPM), що робить цей енкодер підходящим для використання в швидко рухомих механізмах;
- вихідний сигнал A, B, і / або Z сигнали для синхронізації з системами обробки сигналів, можливість використання інкрементного або абсолютного типу виходів.

6) Ступінь захисту IP50 (можливий захист від пилу, але не водонепроникний). Для специфічних умов можуть бути варіанти з вищим рівнем

захисту (IP65). Температурний діапазон зазвичай працює в діапазоні температур від -20°C до +70°C, що робить його стійким до різних умов експлуатації.

7) Використовується в системах керування двигунами для точного вимірювання кута обертання валу або для зворотного зв’язку в роботизованих системах. Застосовується в промислових автоматичних системах для моніторингу позиції в приводах, конвеєрних системах, в робототехніці та інших автоматичних системах. Використовується в машини для різання, пакування, дозування та інших виробничих процесах, де потрібна висока точність позиціонування.

8) Перевага, полягає в тому висока точність завдяки 360 імпульсів на оберт та оптичному принципу роботи, цей енкодер забезпечує високу точність у визначенні кута обертання. Підтримка стандартних інтерфейсів та виходів дозволяє легко інтегрувати енкодер в більші автоматизовані системи. Завдяки конструкції та матеріалам, енкодер є надійним у тривалій експлуатації.

Енкодер Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R є потужним і точним пристроєм для зворотного зв'язку у системах управління. Завдяки своїм характеристикам (360 імпульсів на оберт, висока точність, підтримка різних виходів) він широко використовується в робототехніці, промислових автоматизованих системах і механізмах, де потрібна точна позиційна інформація для регулювання руху або швидкості обертання [8,26].

4. Локальний сервер Raspberry Pi 4 Model B виконує функції обробки даних та управління системою (рис. 3.8).

Raspberry Pi 4 Model B − це потужний одноплатний комп’ютер, який ідеально підходить для використання в ролі локального сервера завдяки своїм характеристикам, можливостям і доступності. Він має більшу потужність порівняно з попередніми моделями і підтримує широкий спектр застосувань, від побутових до промислових. У цьому випадку Raspberry Pi 4 може використовуватися як сервер для зберігання даних, обробки запитів та управління IoT-пристроями.

Основні характеристики Raspberry Pi 4 Model B:

1) Процесор:

– чип Broadcom BCM2711, чотирьохядерний ARM Cortex-A72;
– частота 1.5 GHz;
– архітектура 64-бітна, що дозволяє обробляти складніші операції та виконувати багатозадачність.

2) Оперативна пам'ять (RAM) 2 GB, 4 GB або 8 GB LPDDR4-3200 SDRAM (залежно від модифікації). Вибір об’єму пам'яті дозволяє налаштувати Raspberry Pi для різних завдань − від легких застосувань до складних серверних функцій [14,26].

Рис. 3.8 Локальний сервер Raspberry Pi 4 Model B

3) Графічний процесор VideoCore VI. Порти 2×Micro HDMI порти, підтримка до двох 4K моніторів (4Kp60 або 4Kp30). Підтримує дисплеї з високою роздільною здатністю, що дозволяє використовувати Raspberry Pi як сервер для візуалізації чи моніторингу.

4) Зберігання даних через підключення до microSD слот для карти пам’яті, яка служить основним носієм для ОС і даних, і через USB порти 2 × USB 3.0 порти та 2 × USB 2.0 порти для підключення зовнішніх накопичувачів, таких як жорсткі диски або флешки, та через підключення через мережу Ethernet порту (Gigabit Ethernet), що дозволяє використовувати Raspberry Pi як сервер, підключений до локальної мережі з високою швидкістю [11,24].

5) Вбудований модуль Wi-Fi 802.11ac (2.4GHz та 5GHz), що дозволяє підключати Pi до бездротових мереж. Вбудований Bluetooth 5.0 для підключення різних пристроїв, таких як миші, клавіатури, сенсори або навіть інші Raspberry Pi в систему.

6) Порти вводу/виводу 40 пінів GPIO − розширений набір пінів для підключення периферійних пристроїв, таких як датчики, актуатори, серводвигуни тощо, та інтерфейси I2C, SPI, UART − для зв'язку з різними модулями та сенсорами.

7) Підключення живлення 5V/3A через USB-C порт (більш потужний адаптер для забезпечення стабільної роботи). Споживання енергії в залежності від навантаження, Raspberry Pi 4 споживає від 3W до 15W [13,25].

8) Операційна система Raspberry Pi OS (раніше Raspbian) або інші Linux-системи, такі як Ubuntu Server, що дозволяють налаштувати Raspberry Pi для різних серверних завдань. Можливість запуску Docker контейнерів для ізоляції та керування сервісами. Підтримка серверних додатків, таких як Apache, Nginx, MySQL, Node.js, Python, і багато інших, що дозволяє використовувати Raspberry Pi як локальний сервер для обробки запитів та даних.

9) Плата може потребувати додаткового охолодження (радіаторів або вентиляторів), особливо при високих навантаженнях. Вбудовані механізми захисту від коротких замикань, перевантаження і перегріву.

10) Фізичні розміри 88 мм × 58 мм × 19.5 мм та вага 46 г (без додаткових аксесуарів).

Застосування Raspberry Pi 4 Model B в якості локального сервера:

1) Підтримка програмних середовищ для хостингу вебсайтів (Apache, Nginx, PHP, MySQL), що робить його хорошим вибором для розгортання локальних вебдодатків або тестових середовищ.
2) Використання для зберігання даних, підключення до MySQL, PostgreSQL або SQLite баз даних, що дозволяє зберігати та обробляти інформацію в реальному часі.
3) Raspberry Pi 4 може бути використаний для моніторингу IoT пристроїв, збирання даних від сенсорів та керування іншими пристроями в локальній мережі.
4) Використовується як медіасервер, зокрема для потокового відео або аудіо, через програмне забезпечення, таке як Plex або Kodi.

5) Система для автоматизації (Home Assistant) Raspberry Pi може служити сервером для систем розумного дому, наприклад, для автоматизації управління освітленням, кліматом, безпекою і т. д.
6) Raspberry Pi 4 часто використовується для розробки і тестування IoT-пристроїв і в якості шлюзу для комунікації між пристроями і мережею [14,28].

Переваги Raspberry Pi 4 Model B як локального сервера:

- порівняно з традиційними серверами, Raspberry Pi споживає значно менше енергії;
- малий розмір дозволяє інтегрувати його в обмежені простори;
- недорога ціна для потужного апаратного забезпечення з великою кількістю портів і можливостей;
- великий вибір операційних систем і програмного забезпечення, підтримка розширень і аксесуарів [14,28].

Отже, Raspberry Pi 4 Model B є відмінним варіантом для побудови локального сервера завдяки своїм потужним характеристикам, можливості роботи з різними операційними системами та безлічі можливостей підключення до різних пристроїв та мереж. Це чудовий вибір для малих і середніх проєктів в області автоматизації, розробки IoT, хостингу та моніторингу.

У розділі детально описано структуру апаратної частини системи автоматизації. Запропоновано архітектуру, яка поєднує контролери, сенсори та хмарні сервіси для забезпечення надійного управління електродвигуном. Продемонстровано ефективність обраних компонентів, таких як Raspberry Pi та частотні перетворювачі, що сприяють реалізації функціональних можливостей системи.

РОЗДІЛ 4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Застосування Ignition у системі управління електродвигуном

Ignition by Inductive Automation − це потужна платформа для створення SCADA (Supervisory Control and Data Acquisition) систем, яка дозволяє моніторити, управляти та автоматизувати промислові процеси в реальному часі. Вона підтримує різноманітні типи підключень і протоколів для взаємодії з різними пристроями та системами, включаючи IoT-пристрої, що робить її ідеальною для використання у системах управління електродвигунами на основі IoT [4,12].

Використовуючи наступні компоненти. Ignition комунікує з частотним перетворювачем через стандартні протоколи, такі як Modbus TCP або Modbus RTU (якщо частотний перетворювач підтримує їх), що дозволяє зчитувати параметри роботи двигуна (наприклад, швидкість, струм, напруга) та змінювати параметри роботи через SCADA-систему (рис. 4.1, рис. 4.2).

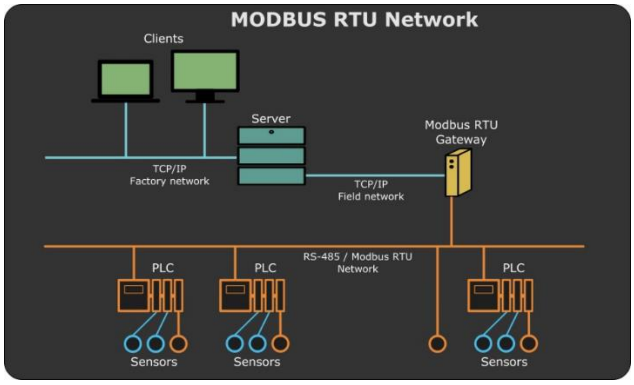


Рис. 4.1 Стандартні протоколи Modbus TCP (Modbus RTU)

У Ignition налаштовуються шаблони, які будуть автоматично коригувати параметри перетворювача в залежності від показників, отриманих з інших пристроїв або за допомогою алгоритмів управління [9,15].

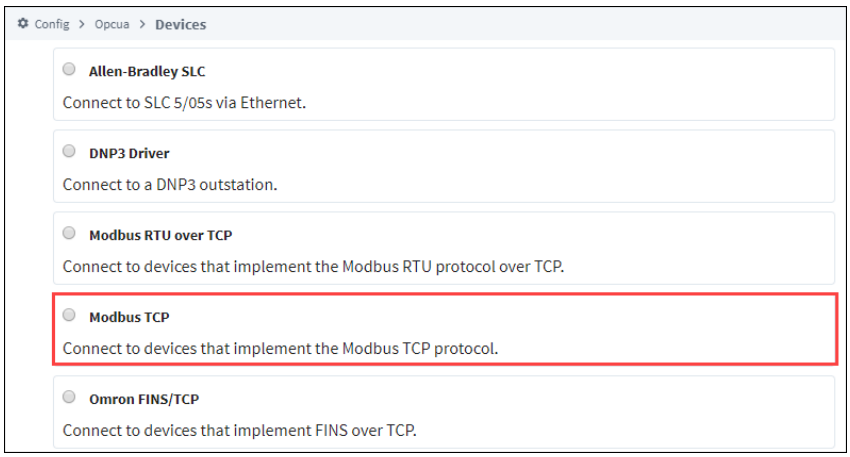


Рис. 4.2 Налаштування протоколу Modbus TCP (Modbus RTU)

Модуль Arduino D1 WiFi (ESP8266 ESP-12E) налаштовується для відправки даних до Ignition через MQTT або OPC-UA, що є зручними протоколами для IoT.

Кроки реалізації:

1) у Arduino IDE встановлюються такі бібліотеки для ESP8266:

- PubSubClient (для MQTT);
- ESP8266WiFi (для WiFi-з'єднання).

2) у коді на Arduino налаштувується підключення до необхідної мережі Wi-Fi;

3) вказується IP-адресу MQTT-брокера (може бути на Ignition або окремий брокер).

Задайте тему (`topic`) для передачі даних (наприклад, `sensors/temperature`).

Відправка даних з датчиків DHT11 (для температури та вологості) та відправляє їх через MQTT (лістинг 4.1).

Лістинг 4.1 Коду для ESP8266 з MQTT

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

// Параметри WiFi
const char* ssid = "Your_WiFi_Name";
const char* password = "Your_WiFi_Password";

// Параметри MQTT
const char* mqtt_server = "192.168.1.100"; // IP-адреса Ignition або брокера
const int mqtt_port = 1883; // Стандартний порт MQTT
```

```
const char* topic = "sensors/temperature";

WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);

  // Підключення до WiFi
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("WiFi connected");

  client.setServer(mqtt_server, mqtt_port);
}

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  // Відправка даних
  float temperature = 25.6; // Замініть на значення з датчика
  char tempString[8];
  dtostrf(temperature, 1, 2, tempString);
  client.publish(topic, tempString);

  delay(5000); // Відправка кожні 5 секунд
}

void reconnect() {
  while (!client.connected()) {
    if (client.connect("ESP8266Client")) {
      Serial.println("Connected to MQTT");
    } else {
      delay(5000);
    }
  }
}
```

У Ignition Gateway встановлюються MQTT Engine (для прийому даних) та MQTT Transmission (для відправки). Підключається Ignition до необхідного MQTT-брокера та вказується на тему `sensors/temperature` у Ignition. Модуль Arduino збирає дані від сенсорів (температура, вібрація, кутове положення) і передає їх у систему для моніторингу та аналізу. Arduino також отримує команди з Ignition для регулювання роботи частотного перетворювача або інших пристроїв в системі. Енкодер, що вимірює кутове положення, може бути підключений до Arduino або безпосередньо до Raspberry Pi через інтерфейси (наприклад, через PWM або інтерфейси, сумісні з Modbus) [4,14].

Raspberry Pi виступає як локальний сервер для збору та обробки даних від сенсорів і пристроїв, а також виконується певна частина аналітики й управління. Він може бути налаштований для виконання локальних алгоритмів, таких як виявлення аномалій чи аналіз ефективності роботи двигуна. Raspberry Pi підключений до Ignition через OPC-UA або MQTT дозволяє передавати дані на сервер або отримувати команди для управління пристроями [6,12]. А саме становлення OPC-UA серверу на Raspberry Pi (рис. 4.3).

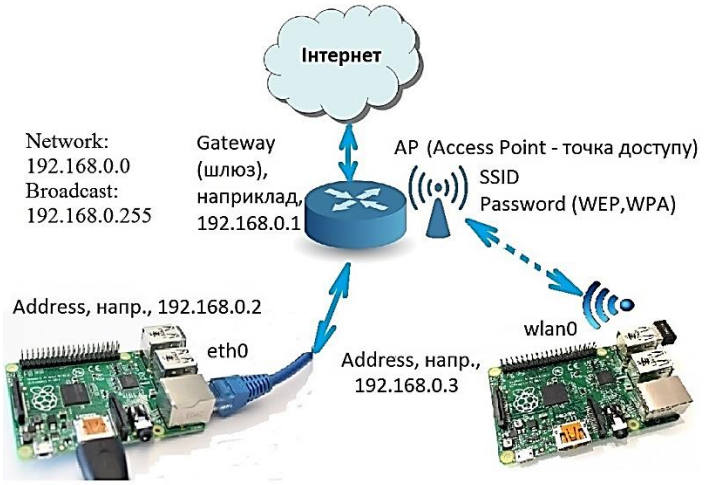


Рис. 4.3 Налаштування підключення Raspberry Pi до мережі

Для цього використовується бібліотека `opcua` для Python, яка дозволяє створювати сервер OPC-UA на Raspberry Pi, яке розписано в лістингу 4.2.

Лістинг 4.2 . Базове налаштування

```
from opcua import Server
import time


server = Server()
server.set_endpoint("opc.tcp://0.0.0.0:4840/freeopcua/server/")

# Додаємо простір імен
uri = "http://example.org"
idx = server.register_namespace(uri)

# Створюємо об'єкти та змінні
objects = server.get_objects_node()
```

```
myvar = objects.add_variable(idx, "MotorEfficiency", 0)
myvar.set_writable()  # дає змогу змінювати значення змінної

server.start()
print("Server started at {}".format(server.endpoint))

try:
    while True:
        # Оновлюємо дані (наприклад, дані ефективності двигуна)
        myvar.set_value(100)  # приклад оновлення
        time.sleep(1)
finally:
    server.stop()
    print("Server stopped")
```

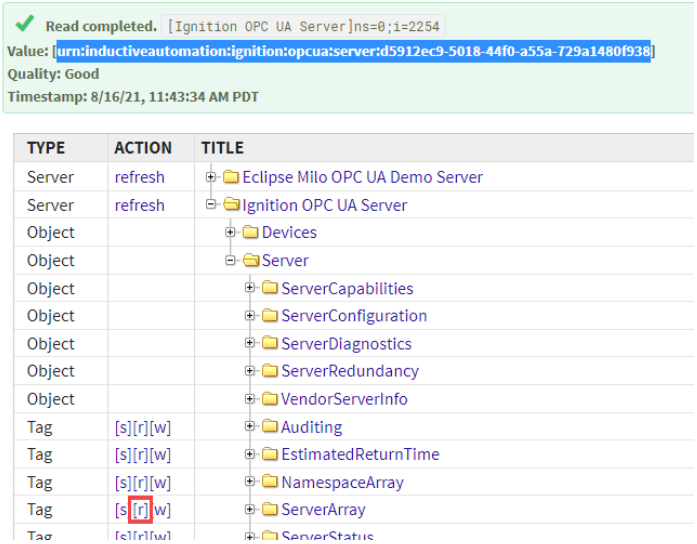


Рис. 4.4 Ignition's OPC UA Server

У Ignition відкривається Gateway Configuration та додається новий OPC-UA сервер (рис. 4.4). Вказується IP-адресу Raspberry Pi і порт (за замовчуванням – 4840), щоб Ignition міг підключитися до OPC-UA сервера на Raspberry Pi. Потім у Ignition налаштовуються зв'язки між змінними (наприклад, для моніторингу ефективності двигуна або інших параметрів). Для передачі даних між Raspberry Pi та Ignition налаштовується MQTT брокер. Найпоширеніший вибір — Mosquitto, який встановлюється на Raspberry Pi за допомогою: `sudo apt-get install mosquitto mosquitto-clients`. Після цього Mosquitto буде працювати як брокер для передачі повідомлень [8,20]. Для передачі даних з Raspberry Pi в MQTT брокер використовується бібліотека `paho-mqtt` для Python (лістинг 4.3).

Лістинг 4.3. Коду для передачі даних:

```
import paho.mqtt.client as mqtt
import time
broker = "localhost"  # або IP-адреса MQTT брокера
topic = "motor/data"
def on_connect(client, userdata, flags, rc):
    print(f"Connected with result code {rc}")
client = mqtt.Client()
client.on_connect = on_connect
client.connect(broker, 1883, 60)
client.loop_start()
try:
    while True:
        # Отправляємо дані про ефективність двигуна
        client.publish(topic, "Motor Efficiency: 95%")
        time.sleep(1)
finally:
    client.loop_stop()
```

Енкодери, такі як Opkon PRI-40, використовуються для вимірювання швидкості обертання або кутових змін. Збір даних з енкодера можна здійснити за допомогою Arduino або іншого мікроконтролера, який підключений до Wi-Fi [11,21]. Спочатку йде зчитування даних у вигляді сигналів з енкодера за допомогою Arduino, які потім передаєте через Wi-Fi, коду для зчитування даних з енкодера (лістинг 4.4).

Лістинг 4.4. Код для зчитування даних з енкодера.

```
const int encoderPinA = 2; // Пін для сигналу A
const int encoderPinB = 3; // Пін для сигналу B

int encoderPos = 0;

void setup() {
  pinMode(encoderPinA, INPUT);
  pinMode(encoderPinB, INPUT);
  attachInterrupt(digitalPinToInterrupt(encoderPinA), readEncoder, CHANGE);
  Serial.begin(115200);
}

void loop() {
  // Можна надсилати дані через WiFi або зберігати на локальному сервері
  Serial.println(encoderPos);
  delay(100);
}

void readEncoder() {
  // Логіка для підрахунку позиції енкодера
  int stateA = digitalRead(encoderPinA);
  int stateB = digitalRead(encoderPinB);
```

```
  if (stateA == stateB) {
    encoderPos++;
  } else {
    encoderPos--;
  }
}
```

Дані з енкодера можна передавати через MQTT або HTTP на сервер для подальшої обробки та відображення на дашборді в Ignition.

4.1 Написання коду для збору даних з датчиків, обробки даних і відправлення їх на сервер

Для створити дашборд в Ignition, який відображає поточну швидкість двигуна, температуру, вібрацію, споживану потужність та історію змін параметрів та були наступні кроками:

Крок 1. Підключення пристроїв.

Для підключення частотного перетворювача DELFA VFD004EL43A використаємо OPC UA для підключення частотного перетворювача до Ignition, що дозволяє отримувати дані про швидкість двигуна та інші параметри.

Далі підключено модуль Arduino D1 WiFi UNO R3 ESP8266 ESP-12E, де використовуються MQTT для передачі даних з сенсорів (температура, вібрація) до Ignition і налаштовується MQTT-брокер в Ignition для прийому даних.

Наступний етап підключення енкодера Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R до контролера, який передаватиме дані в Ignition через цифрові входи або протокол OPC UA.

Далі підключення локального сервера Raspberry Pi 4 Model B, де використовується Ignition Edge на Raspberry Pi для збору та обробки даних локально перед передачею їх у хмару.

Крок 2. Налаштування тегів.

Створення тегів для кожного параметра в Ignition для поточної швидкості двигуна, температури, вібрації, споживаної потужності та інших параметрів. Це дозволить відстежувати ці дані в реальному часі.

У контексті системи Ignition, теги є основою для збору, обробки та відображення даних у реальному часі. Вони дозволяють зв'язати задані параметри з фізичними сенсорами або іншими джерелами даних. Створимо концептуальне уявлення про те, як можна додати теги для параметрів, згаданих у нашому проєкті.

Структура створення тегів у Ignition (у реальному середовищі вона реалізується через Tag Browser або скрипти) розписана у додатку А, де Клас Tag – визначає структуру тегу з полями: ім'я (`name`), тип даних (`data_type`), та значення (`value`), метод `update_value` дозволяє змінювати значення тегу.

Теги створені для кожного параметра:

- швидкість (`current_speed`);
- температура (`current_temperature`);
- вібрація (`current_vibration`);
- потужність (`current_power`);
- історія параметрів (`parameter_history`);

Симуляція даних:

- для тестування значення тегів генеруються випадковим чином;
- значення можна оновлювати в реальному часі за допомогою `update_value`.

Крок 3. Створення дашборду.

Для створення дашборду необхідно запустити середовище Ignition Designer створити новий проєкт або відкрийте існуючий.

Додати компоненти для відображення даних, такі як дисплеї, графіки та таблиці, для відображення поточної швидкості двигуна, температури, вібрації та споживаної потужності та графіки для відображення історії змін параметрів, і зв'язати кожен компонент з відповідним тегом для відображення даних у реальному часі. Налаштування осі графіків та інші параметри для зручного відображення даних.

Крок 4. Налаштування історії даних.

Налаштування історичних тегів проходить через включення історії для тегів, які необхідно відстежувати (рис. 4.5), що дозволить зберігати дані для подальшого аналізу через Tag Browser (`CREATE DATABASE ignition_history;`).

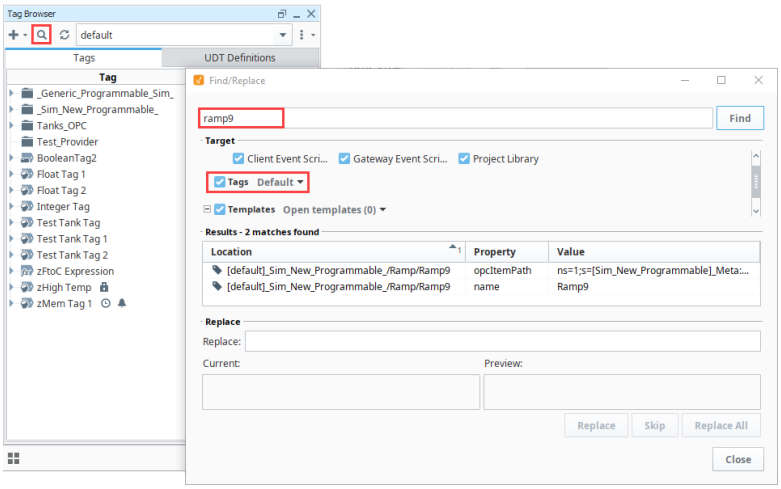


Рис. 4.5 Налаштування аналізу Tag Browser

Для налаштування бази даних необхідно використати базу даних, таку як MySQL або PostgreSQL, для зберігання історичних даних через з'єднання з базою даних в Ignition через Gateway Connection (Ignition Gateway перейдіть до Configuration → Databases → Connections).

Лістинг 4.5 – Налаштування користувача та прав доступу.

```
CREATE USER 'ignition_user'@'%' IDENTIFIED BY 'yourpassword';
GRANT ALL PRIVILEGES ON ignition_history.* TO 'ignition_user'@'%';
FLUSH PRIVILEGES;
```

Даний лістинг дозволяє створення нового користувача, саме через рядки команд:

- `CREATE USER` – створює нового користувача в базі даних.
- `'ignition_user'` – це ім'я нового користувача.
- `'@'%'`– вказує, що користувач може підключатися до бази з будь-якого хосту (відкритий доступ). Якщо потрібно обмежити доступ лише з певного хоста, замість % можна вказати IP-адресу.

– `IDENTIFIED BY 'yourpassword'` – задає пароль для користувача. Замініть 'yourpassword' на ваш власний безпечний пароль.
– `GRANT ALL PRIVILEGES` – надає всі можливі права для роботи з базою даних, включаючи:

  Читання, запис, оновлення та видалення даних.

  Створення, змінення та видалення таблиць або індексів.
– `ON ignition_history.*` – вказує, що права застосовуються до:
– `ignition_history` – ім'я бази даних (замість цього використовуйте ім'я вашої бази, якщо воно інше).
– `.*` – означає всі таблиці цієї бази.
– `TO 'ignition_user'@'%'` – застосовує права до створеного користувача.

Перевірка даних в базі даних

```
SELECT * FROM tag_history_table WHERE tag_id = 'temperature_sensor' LIMIT 10;
```

Крок 5. Візуалізація даних.

Створення дашборду та розміщення компонентів на дашборді для відображення поточних та історичних даних, і налаштування для відображення даних у зручному форматі, щоб дані оновлюються в реальному часі для відображення актуальної інформації. Збір даних з ESP8266. Налаштовується датчик (наприклад, DHT11, DHT22) для вимірювання температури. Відправляти дані через WiFi на сервер або базу даних за допомогою MQTT чи HTTP-протоколу. Використовується база даних (MySQL, PostgreSQL) для збереження температури із часовими мітками. Формат даних: `timestamp`, `temperature`.

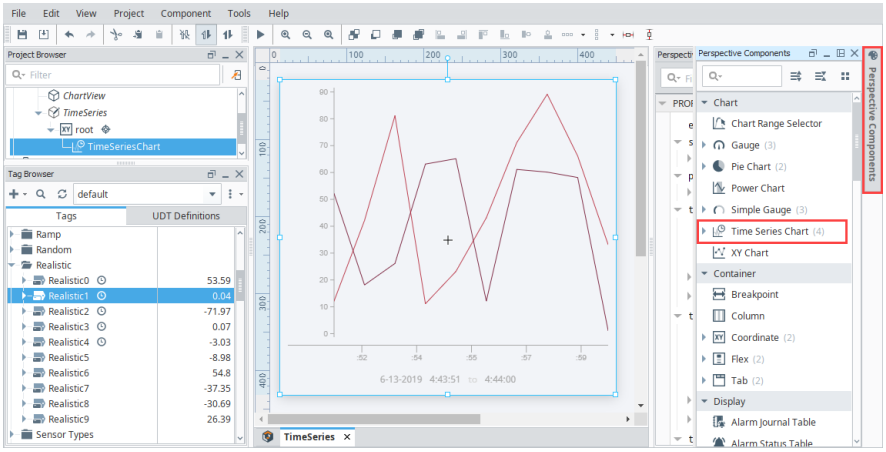


Рис. 4.6 Налаштовується компонент Time Series Chart

Налаштовується компонент Time Series Chart в середовищі Ignition Perspective та підключається джерело даних до бази або тегів (рис. 4.6). Потім налаштовується вісь X для відображення часу, а вісь Y – для температури. Стилізація відбувається через додавання маркерів, кольори, лінії, що відображають тренди та налаштувати діапазони температур для кольорових зон, а саме зелений для нормального рівня, жовтий для помірного, червоний для високого.

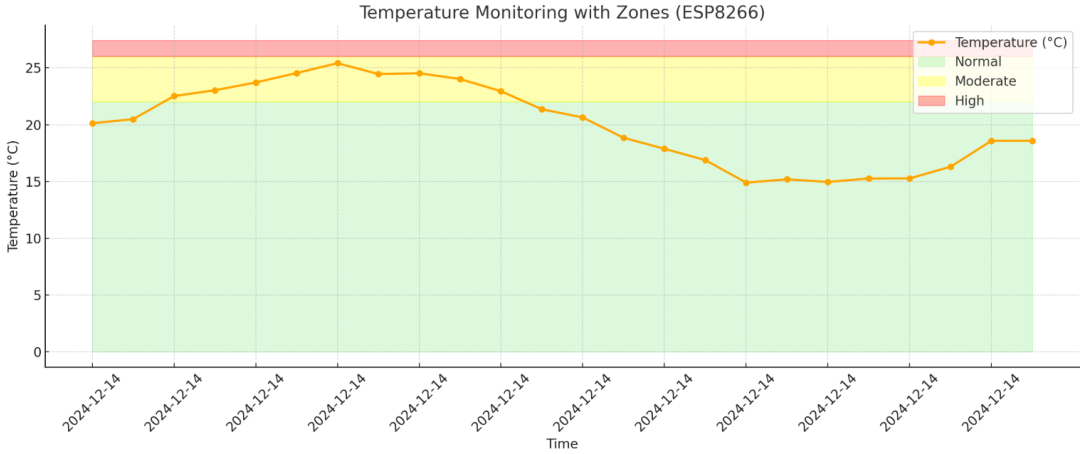


Рис. 4.7 Графік температури з датчика ESP8266

Графік температури з датчика ESP8266 (рис. 4.7), де значення змінюються протягом 24 годин із виділенням зон за рівнями відображений у Ignition Perspective за допомогою Time Series Chart:

1. зелена зона − нормальний рівень температури (до 22°C);
2. жовта зона − помірний рівень (22-26°C);

На рисунку 4.8 представлена графічна ініціалізація даних, що відображає зміни швидкості двигуна, споживання енергії та розподіл статусів обладнання, що дозволяє наочно оцінити ефективність роботи системи.

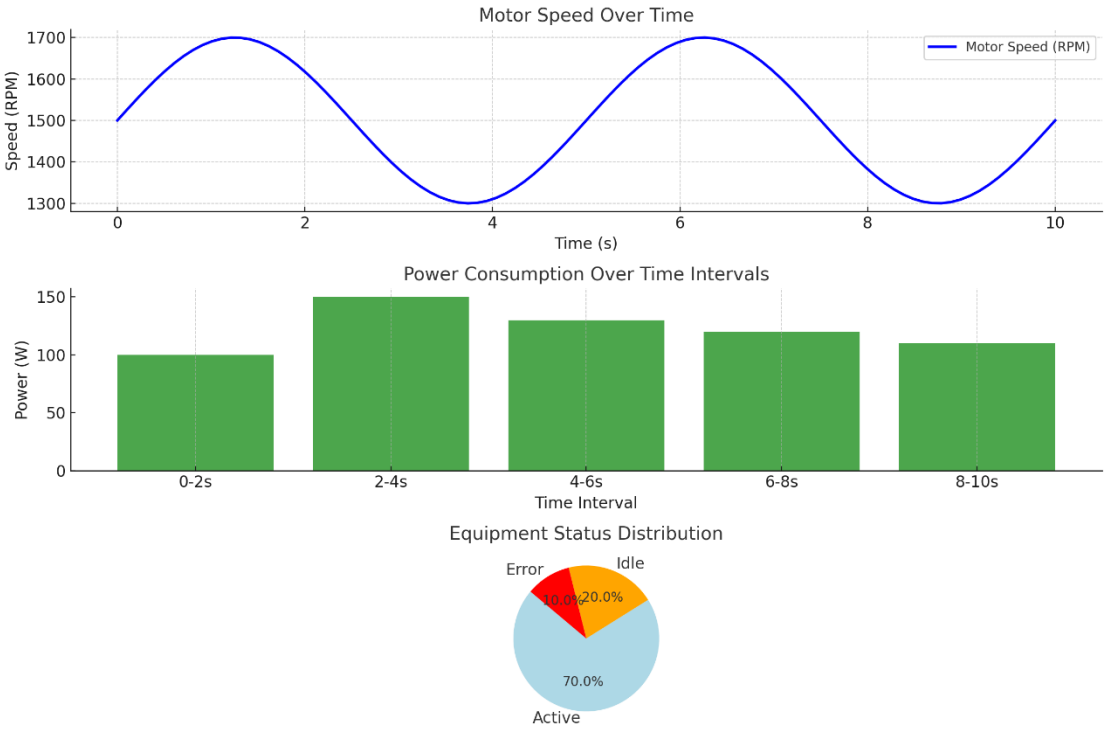


Рис.4.8 Графічна ініціалізація даних зміни швидкості двигуна. споживання енергії, розподіл статусів обладнання.

Три графіки, що відображають дані для запропонованої IoT-архітектури (рис. 4.8):

1. лінійний графік − показує зміну швидкості двигуна (в обертах за хвилину) з часом;
2. стовпчастий графік − відображає споживання енергії за певні часові інтервали;
3. кругова діаграма − візуалізує розподіл статусів обладнання (активний, очікування, помилка).

Крок 6: Публікація дашборду (додаток Б).

Зберегти проєкт в Ignition Designer та опублікування його для доступу через вебінтерфейс. Потім відкрити дашборд через веббраузер для моніторингу та управління системою в реальному часі.

В додатку В у коді реалізовано базовий дашборд для моніторингу роботи двигуна розглянемо ці компоненти:

1. клас Dashboard − використовується для створення головного контейнера дашборда та містить методи:
   `add_widget` − додає віджети (графіки чи таблиці) до дашборда;
   `render` − генерує текстове представлення всіх віджетів дашборда.
2. клас DataSource − симулює джерело даних для отримання параметрів, таких як швидкість, температура, вібрація тощо, та метод `get_data` повертає випадкові значення, щоб замінити реальні дані для тестування.
3. клас Graph − використовується для представлення даних у вигляді графіків (тип графіка визначається аргументом `graph_type`) та метод `render` формує текстове представлення графіка з останніми даними.
4. клас Table − використовується для відображення даних у табличному вигляді таметод `render` генерує список значень для табличного представлення даних.
5. створення дашборда − ініціалізується дашборд із назвою "Motor Performance Monitoring" (рис. 4.9). Визначаються джерела даних для кожного параметра двигуна (швидкість, температура тощо). Додаються відповідні віджети до дашборда.
6. виведення дашборда − метод `render` повертає текстове представлення всіх графіків і таблиць із даними.

Дані кроки допомогли створити дашборд в Ignition, який відображатиме всі необхідні параметри для управління електродвигуном із запропонованою архітектурою IoT.

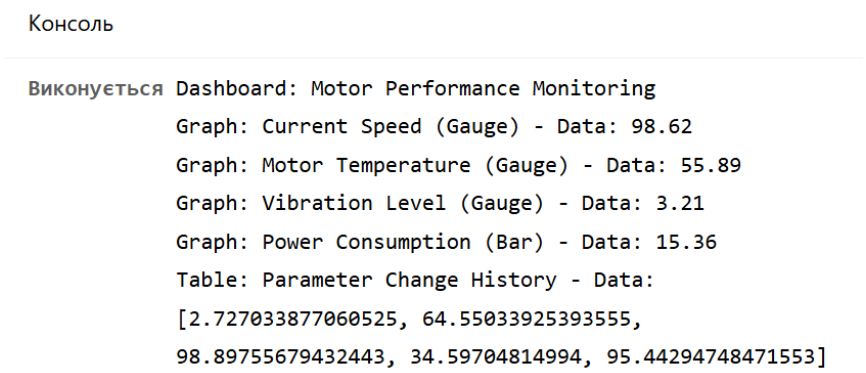


Рис. 4.9 Виконання "Motor Performance Monitoring"

Мета коду − це навчальний приклад, який дозволяє симулювати моніторинг двигуна. Реальні запити до бази даних або сенсорів (як у системі IoT) тут замінено на генерацію випадкових даних для тестування.

Якщо необхідно інтегрувати реальні джерела даних, потрібно замінити метод get_data у DataSource на логіку для підключення до бази даних або сенсорів. Забезпечити обробку реальних даних замість випадкових значень [19,30].

У розділі описано процес розробки програмного забезпечення для збору, обробки та аналізу даних у системі управління. Використання інструментів, таких як Ignition та хмарні платформи AWS/Azure, дозволило створити зручний інтерфейс для моніторингу роботи електродвигуна, забезпечивши інтеграцію з аналітичними сервісами для підвищення ефективності.

ВИСНОВКИ

У магістерській роботі розроблено систему автоматизації промислових процесів на базі мікропроцесорних систем та IoT. Аналіз існуючих рішень показав перспективність використання цих технологій.

У першому розділі «Аналіз існуючих рішень у сфері автоматизації промислових процесів та IoT» проведено ґрунтовний аналіз сучасних технологій автоматизації промислових процесів та Інтернету речей (IoT). Встановлено, що інтеграція традиційних систем автоматизації (SCADA, PLC, DCS) із технологіями IoT суттєво підвищує ефективність промислових операцій, забезпечуючи більш точний моніторинг, аналіз даних у реальному часі та прийняття обґрунтованих рішень [1,12].

Ключові аспекти розгляду включають:

Традиційні системи автоматизації як SCADA забезпечує централізований моніторинг і управління, PLC виконує локальне керування із високою адаптивністю до змін процесів, DCS інтегрує децентралізоване управління для великих систем, забезпечуючи їхню надійність.

Інтернет речі (IoT) розглядають роль сенсорних мереж для збору параметрів, таких як температура, вологість, вібрація тощо. Виділено протоколи зв’язку (MQTT, CoAP, LoRaWAN), які забезпечують ефективну передачу даних. Наголошено на значенні IoT-платформ (AWS IoT, Azure IoT, Node-RED) для обробки даних, віддаленого управління та аналітики [3,15].

Мікропроцесорні системи розглядають застосування Arduino, Raspberry Pi, STM32 та MSP430 у промисловій автоматизації. Зроблено порівняння їх функціональності, продуктивності та сфери використання.

Інтеграція IoT із мікропроцесорними системами підкреслено, що поєднання цих технологій сприяє створенню інтелектуальних систем управління з можливістю обробки великих обсягів даних, прогнозування несправностей та оптимізації ресурсів.

Сучасні фреймворки для IoT Node-RED відзначено за простоту інтеграції. AWS IoT та Azure IoT підходять для складних і масштабованих проєктів із високими вимогами до безпеки [8,23].

Результати аналізу демонструють, що використання IoT у поєднанні з мікропроцесорними системами створює значний потенціал для автоматизації та оптимізації промислових процесів. Такий підхід дозволяє зменшити витрати, мінімізувати час простоїв, підвищити ефективність виробництва та забезпечити сталий розвиток підприємств. Отримані результати є основою для подальшої розробки інтегрованих систем автоматизації.

У другому розділі "Розробка концепції автоматизації" розглянуто концепцію автоматизації промислового електродвигуна із використанням технологій Інтернету речей (IoT). Розробка орієнтована на інтеграцію апаратних і програмних компонентів для забезпечення ефективного моніторингу, управління та аналізу параметрів роботи двигуна у реальному часі.

Концепція автоматизації запропоновано принципи інтеграції IoT-пристроїв, включаючи використання сенсорів, протоколів передачі даних (MQTT, OPC-UA) і хмарних платформ (AWS IoT Core, Azure IoT Hub). Реалізовано архітектуру системи, яка передбачає локальне управління на базі контролера Arduino та сервера Raspberry Pi для обробки даних, а також хмарну інтеграцію для довготривалого зберігання та аналізу даних. Розроблено детальний алгоритм, що включає збір, обробку та аналіз даних сенсорів, формування керуючих сигналів для частотного перетворювача, а також прогнозування несправностей за допомогою машинного навчання. Запропоновано механізми інтерактивної взаємодії користувача із системою через вебінтерфейс, повідомлення про критичні ситуації та дистанційне управління. Система забезпечує адаптивність до змін умов експлуатації шляхом динамічного налаштування параметрів роботи двигуна. Використання IoT-підходів дозволяє мінімізувати час простоїв, знизити енергоспоживання та оптимізувати процеси технічного обслуговування [8,24].

Реалізація запропонованої концепції відкриває широкі можливості для підвищення ефективності виробництва. Її впровадження сприятиме зниженню

витрат на обслуговування обладнання, підвищенню надійності та продуктивності, а також зростанню конкурентоспроможності підприємства. Подальші дослідження можуть бути спрямовані на розширення функціоналу системи шляхом інтеграції додаткових сенсорів, впровадження розширених алгоритмів машинного навчання та використання інших хмарних платформ для більш гнучкого управління [7,23].

У третьому розділі «Проєктування апаратної частини» детально розглянуто апаратну складову системи автоматизації управління електродвигуном на основі IoT, яка забезпечує гнучкість, адаптивність та ефективність роботи в промислових умовах.

Апаратна архітектура запропоновано інтеграцію частотного перетворювача DELFA VFD004EL43A, модуля Arduino D1 WiFi UNO R3 ESP8266, енкодера Opkon PRI-40-A-R6-HLD-360-ZZ-V3-2M5-R та локального сервера Raspberry Pi 4 Model B. Така система поєднує функції управління, моніторингу й обробки даних, зокрема за допомогою хмарних платформ Azure та AWS. Використання частотного перетворювача забезпечує точне регулювання швидкості обертання двигуна і його захист від перевантажень. Модуль Arduino з вбудованими можливостями Wi-Fi дозволяє організувати бездротовий зв’язок, що спрощує управління та налаштування. Енкодер Opkon забезпечує високу точність вимірювання швидкості та положення обертання, сприяючи коректному функціонуванню системи. Raspberry Pi виступає локальним сервером для обробки даних і управління пристроями, інтегруючись із хмарними сервісами для довгострокового зберігання та аналізу інформації. AWS Cloud та Azure IoT забезпечують безпеку, масштабованість і додаткові можливості для аналізу даних, що дозволяє виявляти несправності, прогнозувати технічне обслуговування та оптимізувати роботу електродвигуна.

Система дозволяє підключати додаткові датчики (температури, вібрації), налаштовувати параметри в реальному часі та інтегрувати нові компоненти відповідно до вимог[5,27].

Представлену систему можна використовувати в промислових установках, будівельній техніці, роботизованих механізмах тощо. Її перевагами є економія енергії, підвищення точності управління та надійності роботи.

Запропонована система управління електродвигуном із використанням IoT є оптимальним рішенням для автоматизації в промислових умовах, оскільки поєднує сучасні апаратні засоби з потужними можливостями хмарних сервісів. Обрані компоненти демонструють високу функціональність, гнучкість та адаптивність, сприяючи ефективному управлінню, моніторингу й аналізу даних у реальному часі [21,23].

У четвертому розділі «Розробка програмного забезпечення» присвячено розробці програмного забезпечення для автоматизації процесів управління електродвигуном у системі IoT. Було реалізовано комплексний підхід до збору, обробки, аналізу даних і візуалізації результатів у реальному часі, що забезпечило високу функціональність і зручність використання системи.

Платформа Ignition стала основою для створення SCADA-системи з підтримкою стандартних протоколів (MQTT, OPC-UA), що забезпечило інтеграцію всіх компонентів системи, включаючи датчики, контролери та хмарні сервіси.

Було створено дашборд для відображення ключових параметрів роботи електродвигуна (швидкість, температура, вібрація, споживана потужність), а також графіки змін у режимі реального часу та історичних даних. Використано мікроконтролери Arduino D1 WiFi (ESP8266 ESP-12E) для збору даних з датчиків і передачі їх через MQTT. Локальний сервер Raspberry Pi 4 Model B обробляє дані та забезпечує їх передачу на платформу Ignition через протоколи OPC-UA та MQTT. Енкодери Opkon PRI-40 інтегровані для вимірювання швидкості обертання двигуна, а також для забезпечення зворотного зв’язку.

Написані програми для Arduino і Raspberry Pi для збирання даних з датчиків та енкодерів, їх обробки й передачі на сервер. Використання Python та бібліотек (paho-mqtt, opcua) для розробки серверної частини, яка інтегрується із системою Ignition. Реалізація функцій аналізу даних, виявлення аномалій і автоматичного

коригування параметрів двигуна. Система підключена до хмарних платформ AWS та Azure для довгострокового зберігання даних, їх аналізу та використання прогнозного обслуговування. Використання аналітичних інструментів і візуалізація даних через вебінтерфейс дозволили забезпечити доступність і зручність у використанні. Створено зручний дашборд для моніторингу роботи електродвигуна, що включає ключові параметри та інструменти для аналізу. Забезпечено інтеграцію з сенсорами та хмарними платформами, що дозволяє ефективно управляти обладнанням, оперативно реагувати на зміни умов та забезпечувати безперебійність роботи системи. Реалізована система підтримує масштабованість, гнучкість і можливість інтеграції з додатковими сенсорами й пристроями, що робить її універсальним рішенням для промислової автоматизації [6,27].

Розробка програмного забезпечення успішно реалізувала концепцію IoT у системі управління електродвигуном. Використані інструменти й технології дозволили досягти високої ефективності, інтеграції й автоматизації, відкриваючи можливості для подальшого вдосконалення системи та її адаптації до різних галузей промисловості.

На підставі отриманих результатів зроблено висновок, що впровадження систем автоматизації на базі IoT забезпечує економію ресурсів, зниження експлуатаційних витрат та підвищення конкурентоспроможності підприємств.

Результати дослідження підтверджують доцільність використання IoT-технологій для автоматизації промислових процесів. Запропоновані рішення мають практичну цінність і можуть бути впроваджені на сучасних підприємствах для досягнення максимальної ефективності.

ПЕРЕЛІК ПОСИЛАНЬ

1. 5 Ways IoT is Revolutionizing Industrial Automation. [Електронний ресурс]. Режим доступу: https://www.simbase.com/blog/5-ways-iot-is-revolutionizing-io
2. AI, IoT & Automation: Revolutionizing Electric Motors: [Електронний ресурс]. Режим доступу: https://iecmotores.com/ai-iot-automation-electric-motors/
3. Benefits of Modernizing On-premises Analytics with an AWS Lake House. [Електронний ресурс]. Режим доступу: https://aws.amazon.com/blogs/architecture/benefits-of-modernizing-on-premise-analytics-with-an-aws-lake-house/
4. How Are AI, IoT, and Automation Transforming Electric Motors? [Електронний ресурс]. Режим доступу: https://iecmotores.com/ai-iot-automation-electric-motors/
5. Ignition's OPC UA Server. [Електронний ресурс]. Режим доступу: https://www.docs.inductiveautomation.com/docs/8.1/ignition-modules/opc-ua/ignitions-opc-ua-server
6. IIoT-ready technologies improve machine controls. [Електронний ресурс]. Режим доступу: https://www.controleng.com/articles/iiot-ready-technologies-improve-machine-controls/
7. Internet of Things, IoT. [Електронний ресурс]. Режим доступу: https://www.it.ua/knowledge-base/technology-innovation/internet-veschej-internet-of-things-iot
8. IOT Controlled Industrial Automation. [Електронний ресурс]. Режим доступу: https://www.researchgate.net/publication/355486714_IOT_Controlled_Industrial_Automation
9. IoT панелі управління: ключові вимоги та функції для моніторингу пристроїв. [Електронний ресурс]. Режим доступу: https://wezom.com.ua/ua/blog/iot-paneli-upravlinnya

10. IoT системи програмне забезпечення сервіс датчики. [Електронний ресурс]. Режим доступу: https://radmirtech.com.ua/wp-content/uploads/2021/04/kataloh_voda_28_04_21.pdf
11. Low-Cost Industrial IoT System for Wireless Monitoring of Electric Motors Condition. [Електронний ресурс]. Режим доступу: https://link.springer.com/article/10.1007/s11036-022-02017-2
12. Machine Learning and IoT-Based Solutions in Industrial Applications for Smart Manufacturing: A Critical Review. [Електронний ресурс]. Режим доступу: https://www.mdpi.com/1999-5903/16/11/394
13. Make your machines IIoT-ready with smart motor control solutions. [Електронний ресурс]. Режим доступу: https://www.controleng.com/articles/iiot-ready-technologies-improve-machine-controls/
14. Real-Time Monitoring of Electric Motors for Detection of Operating Anomalies and Predictive Maintenance. [Електронний ресурс]. Режим доступу: https://www.researchgate.net/publication/343235823_Real-Time_Monitoring_of_Electric_Motors_for_Detection_of_Operating_Anomalies_and_Predictive_Maintenance
15. Tag Browser. [Електронний ресурс]. Режим доступу: https://www.docs.inductiveautomation.com/docs/8.1/platform/tags/tag-browser.
16. The Role of IoT in Industrial Automation in 2024: У цій статті обговорюється важлива роль IoT в промисловій автоматизації, включаючи підвищення продуктивності та зменшення простоїв. [Електронний ресурс]. Режим доступу: https://www.toobler.com/blog/role-of-iot-in-industrial-automation
17. What is Cloud Architecture? Understanding the Building Blocks of the Cloud. [Електронний ресурс]. Режим доступу: https://www.digitalocean.com/resources/articles/cloud-architecture
18. Використання сучасних комп’ютерних технологій для автоматизації виробничих процесів. [Електронний ресурс]. Режим доступу:

https://media.neliti.com/media/publications/311344-using-modern-computer-technology-for-aut-d5c31956.pdf

19. Дізнайтеся про архітектуру AWS детально з найкращими практиками. [Електронний ресурс]. Режим доступу: https://www.projectpro.io/article/aws-architecture-with-diagram/575
20. Дослідження системи моніторингу та управління елементами IoT. [Електронний ресурс]. Режим доступу: https://er.nau.edu.ua/bitstream/NAU/62635/1/ФКНТ_2023_123_Корінний Е.pdf
21. Інтернет речей (IoT) – що це таке і як працює, суть, технології і приклади. [Електронний ресурс]. Режим доступу: https://termin.in.ua/internet-rechey-iot/
22. Мікропроцесорні засоби в автоматизованих системах керування технологічними процесами. [Електронний ресурс]. Режим доступу: https://repository.kpi.kharkov.ua/items/c9f7cc3d-0ad8-4d60-95b5-defe7a34ad97
23. Мікропроцесорні системи автоматизації. [Електронний ресурс]. Режим доступу: https://moodle.znu.edu.ua/pluginfile.php/1031033/md_rerce/tet/1/МПС_А.pdf
24. Моніторинг та управління елементами IoT. [Електронний ресурс]. Режим доступу: https://er.nau.edu.ua/bm/U/426/1/ПІ_123_2020_ШевченкоО.О.pdf
25. Налаштування підключення Raspberry Pi до мережі. [Електронний ресурс]. Режим доступу: https://mikrotik.kpi.ua/index.php/courses-list/category-raspberry/68-configuring-the-connection-to-the-raspberry-pi-session-3
26. Організація системи забезпечення єдиного часу в IoT. [Електронний ресурс]. Режим доступу: https://conf.ztu.edu.ua/wp-content/uploads/2020/05/7.-modelyuvannya-ta-programuvannya-informatsijnyh-system.pdf
27. Промисловий Інтернет речей: що потрібно знати про сучасну технологію? [Електронний ресурс]. Режим доступу: https://pnn.com.ua/ua/blog/detail/industrial-internet-of-things-iiot-up-to-date-solutions-and-keys-you-need-to-know

28. Розкрийте вбудовані системи IoT: прискорення розробки вашого IoT. [Електронний ресурс]. Режим доступу: https://www.dusuniot.com/uk/blog/unveil-iot-embedded-system/
29. Система збору та аналізу даних для IoT платформ. [Електронний ресурс]. Режим доступу: https://er.knutd.edu.ua/bitstream/123456789/21047/1/Dyplom123_StatsenkoV_Osypenko.pdf
30. Системи промислової автоматизації на основі IoT. [Електронний ресурс]. Режим доступу: http://gbdmm.knuba.edu.ua/article/view/232470

Демонстраційні матеріали (презентація):

Державний університет інформаційно-комунікаційних технологій
Кафедра Інформаційних систем та технологій

СИСТЕМА АВТОМАТИЗАЦІЇ ПРОМИСЛОВИХ ПРОЦЕСІВ НА БАЗІ МІКРОПРОЦЕСОРНИХ СИСТЕМ З ІНТЕРНЕТОМ РЕЧЕЙ

на здобуття освітнього ступеня магістра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

ВИКОНАВ: ЗДОБУВАЧ ВИЩОЇ ОСВІТИ ГР. ІСДМ-61
АНДРІЙ ЗАЯЧКОВСЬКИЙ
КЕРІВНИК: ІГОР СІНІЦИН

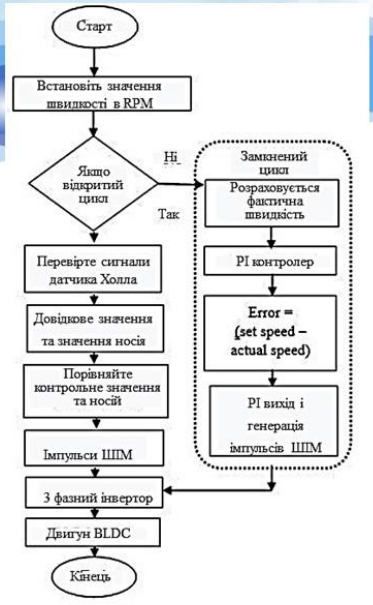
Об'єкт дослідження: промислові процеси, що можуть бути автоматизовані за допомогою інтегрованих IoT та мікропроцесорних систем.

Предмет дослідження методи та засоби інтеграції мікропроцесорних систем з технологіями Інтернету речей для автоматизації промислових процесів, а також алгоритми управління та аналізу даних для підвищення ефективності автоматизованої системи.

Метою магістерської роботи є дослідження методів інтеграції мікропроцесорних систем з Інтернетом речей для автоматизації промислових процесів.

Для досягнення цієї мети необхідно вирішити ***наступні завдання*:**

- провести аналіз сучасного стану технологій IoT та мікропроцесорних систем у контексті автоматизації промислових процесів;
- розробити архітектуру інтегрованої IoT-системи для автоматизації промислових процесів, включаючи вибір компонентів та протоколів зв'язку;
- реалізувати прототип інтегрованої системи та протестувати її в умовах, наближених до реальних виробничих процесів;
- провести оцінку ефективності розробленої системи та порівняти її з існуючими рішеннями для автоматизації на базі IoT та мікропроцесорів.

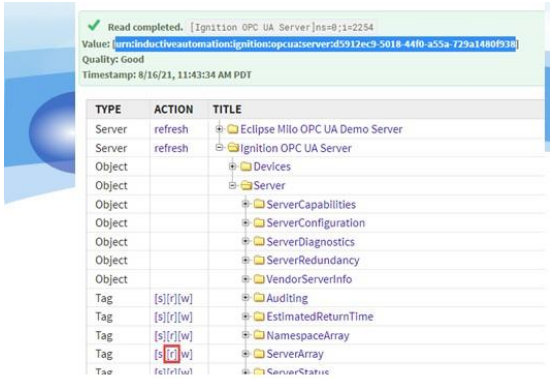


Блок-схема для регулювання швидкості приводу двигуна

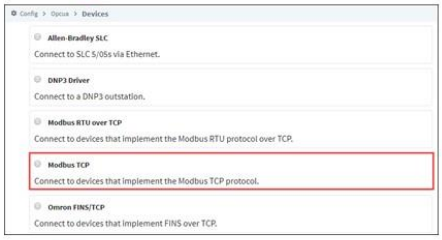
У другому розділі "Розробка концепції автоматизації" розглянуто концепцію автоматизації промислового електродвигуна із використанням технологій Інтернету речей (IoT). Розробка орієнтована на інтеграцію апаратних і програмних компонентів для забезпечення ефективного моніторингу, управління та аналізу параметрів роботи двигуна у реальному часі.

Концепція автоматизації запропоновано принципи інтеграції IoT-пристроїв, включаючи використання сенсорів, протоколів передачі даних (MQTT, OPC-UA) і хмарних платформ (AWS IoT Core, Azure IoT Hub).

У першому розділі «Аналіз існуючих рішень у сфері автоматизації промислових процесів та IoT» проведено ґрунтовний аналіз сучасних технологій автоматизації промислових процесів та Інтернету речей (IoT). Встановлено, що інтеграція традиційних систем автоматизації (SCADA, PLC, DCS) із технологіями IoT суттєво підвищує ефективність промислових операцій, забезпечуючи більш точний моніторинг, аналіз даних у реальному часі та прийняття обґрунтованих рішень.

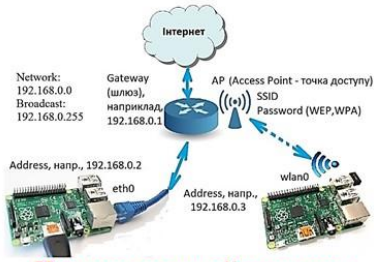


Ignition's OPC UA Server



Налаштування протоколу Modbus TCP (Modbus RTU)

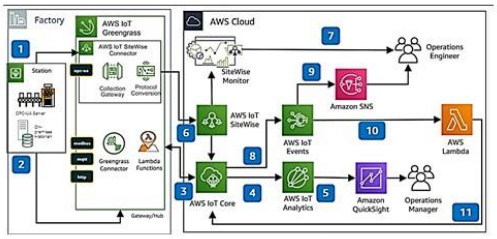
У четвертому розділі «Розробка програмного забезпечення» присвячено розробці програмного забезпечення для автоматизації процесів управління електродвигуном у системі IoT. Було реалізовано комплексний підхід до збору, обробки, аналізу даних і візуалізації результатів у реальному часі, що забезпечило високу функціональність і зручність використання системи.



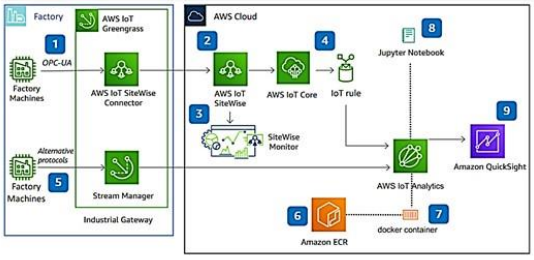
Налаштування підключення Raspberry Pi до мережі

Запропонована система автоматизації промислового електродвигуна засобами з IoT

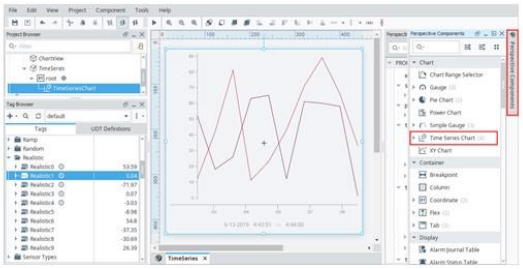
У третьому розділі «Проєктування апаратної частини» детально розглянуто апаратну складову системи автоматизації управління електродвигуном на основі IoT, яка забезпечує гнучкість, адаптивність та ефективність роботи в промислових умовах.



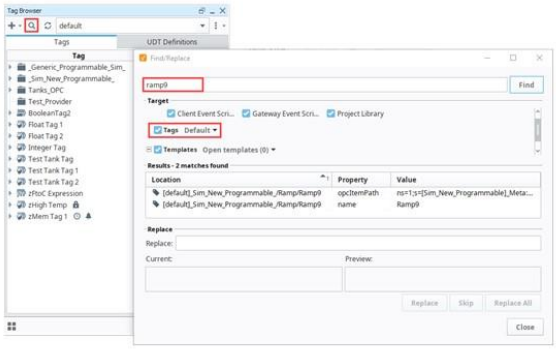
Архітектура системи AWS Cloud моніторингу ресурсів



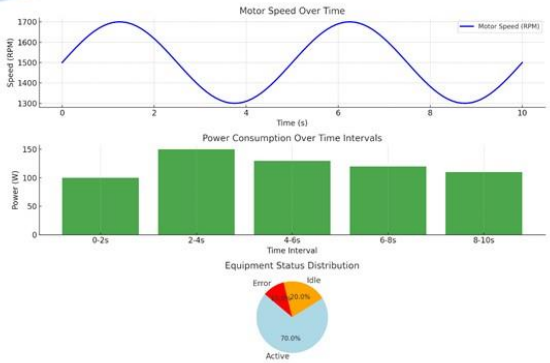
Архітектура системи AWS Cloud аналітики технічного обслуговування



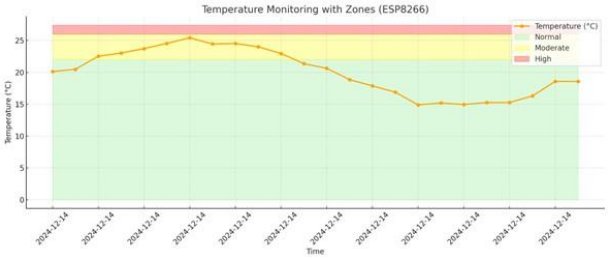
Налаштовується компонент Time Series Chart



Налаштування аналізу Tag Browser



Графічна ініціалізація даних зміни швидкості двигуна, споживання енергії, розподіл статусів обладнання



Графік температури з датчика ESP8266

ВИСНОВКИ

На підставі отриманих результатів зроблено висновок, що впровадження систем автоматизації на базі IoT забезпечує економію ресурсів, зниження експлуатаційних витрат та підвищення конкурентоспроможності підприємств.

Результати дослідження підтверджують доцільність використання IoT-технологій для автоматизації промислових процесів. Запропоновані рішення мають практичну цінність і можуть бути впроваджені на сучасних підприємствах для досягнення максимальної ефективності.

Апробація роботи

Заячковський А. В., Завацький В.С. «МОНІТОРИНГ ТИ ВИКОРИСТАННЯ IoT ДЛЯ АВТОМАТИЗАЦІЇ ПРОМИСЛОВИХ ПРОЦЕСІВ». Стаття у загальногалузевому науково-виробничому журналі «Зв'язок», м.Київ - №6, 2024. – С. 81-92

Заячковський А. В. «МОНІТОРИНГ ТИ ВИКОРИСТАННЯ IoT ДЛЯ АВТОМАТИЗАЦІЇ ПРОМИСЛОВИХ ПРОЦЕСІВ». Тези у II всеукраїнська науково-технічна конференція «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу» – Київ, 16 січня 2024 р.

Додаток А. Структура створення тегів у Ignition

```
    class Tag:
        def __init__(self, name, data_type, value):
            self.name = name
            self.data_type = data_type
            self.value = value

        def update_value(self, new_value):
            self.value = new_value

        def __repr__(self):
            return  f"Tag:  {self.name},  Type:  {self.data_type},  Value:
{self.value}"

    # Create tags for each parameter
    tags = {
        "current_speed":    Tag(name="current_speed",    data_type="Float",
value=random.uniform(0, 100)),
        "current_temperature":             Tag(name="current_temperature",
data_type="Float", value=random.uniform(20, 80)),
        "current_vibration":                 Tag(name="current_vibration",
data_type="Float", value=random.uniform(0, 10)),
        "current_power":    Tag(name="current_power",    data_type="Float",
value=random.uniform(100, 1000)),
        "parameter_history":                 Tag(name="parameter_history",
data_type="List", value=[]),
    }

    # Simulate updating tag values
    tags["current_speed"].update_value(75.5)
    tags["current_temperature"].update_value(55.0)
    tags["current_vibration"].update_value(2.5)
    tags["current_power"].update_value(450.0)
    # Display tags
    for tag in tags.values():
        print(tag)
```

Додаток Б. Ignition Dashboard Motor

```
1  import random
2
3  class Dashboard:
4      def __init__(self, title):
5          self.title = title
6          self.widgets = []
7
8      def add_widget(self, widget):
9          self.widgets.append(widget)
10
11     def render(self):
12         dashboard_representation = f"Dashboard: {self.title}\n"
13         for widget in self.widgets:
14             dashboard_representation += widget.render() + "\n"
15         return dashboard_representation
16
17 class DataSource:
18     def __init__(self, name, query):
19         self.name = name
20         self.query = query
21
22     def get_data(self):
23         # Simulating data retrieval with random values for the sake of
   this example
24         return random.uniform(0, 100)
25
26 class Graph:
27     def __init__(self, data_source, title, graph_type):
28         self.data_source = data_source
29         self.title = title
30         self.graph_type = graph_type
31
32     def render(self):
33         data = self.data_source.get_data()
34         return  f"Graph:  {self.title}  ({self.graph_type})  -  Data:
   {data:.2f}"
35
36 class Table:
37     def __init__(self, data_source, title):
38         self.data_source = data_source
39         self.title = title
40
41     def render(self):
42         data  =  [self.data_source.get_data()  for  _  in  range(5)]    #
   Simulating table rows
43         return f"Table: {self.title} - Data: {data}"
44
45 # Create a new Ignition dashboard
46 dash = Dashboard(title="Motor Performance Monitoring")
47
48 # Define data sources
49 speed_data  =  DataSource("current_speed",  query="SELECT  speed  FROM
   motor_data ORDER BY timestamp DESC LIMIT 1")
50 temp_data = DataSource("current_temperature", query="SELECT temperature
   FROM motor_data ORDER BY timestamp DESC LIMIT 1")
```

```
vibration_data = DataSource("current_vibration", query="SELECT vibration FROM motor_data ORDER BY timestamp DESC LIMIT 1")
power_data = DataSource("current_power", query="SELECT power FROM motor_data ORDER BY timestamp DESC LIMIT 1")
history_data = DataSource("parameter_history", query="SELECT * FROM motor_data ORDER BY timestamp DESC LIMIT 100")

# Add widgets to the dashboard
# Speed
speed_graph = Graph(data_source=speed_data, title="Current Speed", graph_type="Gauge")
dash.add_widget(speed_graph)

# Temperature
temp_graph = Graph(data_source=temp_data, title="Motor Temperature", graph_type="Gauge")
dash.add_widget(temp_graph)

# Vibration
vibration_graph = Graph(data_source=vibration_data, title="Vibration Level", graph_type="Gauge")
dash.add_widget(vibration_graph)

# Power Consumption
power_graph = Graph(data_source=power_data, title="Power Consumption", graph_type="Bar")
dash.add_widget(power_graph)

# Parameter History
history_table = Table(data_source=history_data, title="Parameter Change History")
dash.add_widget(history_table)

# Render the dashboard
print(dash.render())
```