НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет електроніки

Кафедра мікроелектроніки

«На правах рукопису»
УДК 004.42:57.08

До захисту допущено:

Завідувач кафедри

_____ Дмитро ТАТАРЧУК
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Мікро- та наноелектроніка»

зі спеціальності 176 «Мікро- та наносистемна техніка»

на тему: «Програмне забезпечення користувача системи вимірювання лабораторії на чипі»

Виконав:
студент ІІ курсу, групи ДП-41мп
Розум Євген Олександрович _____

Науковий керівник: доц.каф.МЕ, к.ф.-м.н., с.н.с.,
Свєчніков Георгій Сергійович _____

Консультант з нормоконтролю: доц.каф.МЕ, к.ф.-м.н., с.н.с.,
Свєчніков Георгій Сергійович _____

Консультант з інформаційних питань: доц.каф.МЕ, к.т.н., доц.,
Діденко Юрій Вікторович _____

Рецензент: В.о. зав.каф.ЕІ, к.т.н., доц.,
Казміренко Віктор Анатолійович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет електроніки
Кафедра мікроелектроніки

Рівень вищої освіти – другий (магістерський)
Спеціальність – 176 «Мікро- та наносистемна техніка»
Освітньо-професійна програма «Мікро- та наноелектроніка»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Дмитро ТАТАРЧУК
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Розуму Євгену Олександровичу

1. Тема дисертації «Програмне забезпечення користувача системи вимірювання лабораторії на чипі», науковий керівник дисертації Свєчніков Георгій Сергійович, к.ф.-м.н., с.н.с., затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом дисертації
01.12.2025
3. Об’єкт дослідження
Процес автоматизованого моніторингу та керування фізико-хімічними параметрами в мікрофлюїдних системах культивування клітин.
4. Вихідні дані
   1. Технічні вимоги до системи: необхідність забезпечення дистанційного моніторингу в реальному часі, використання бездротових технологій (Wi-Fi).
   2. Характеристики апаратної частини лабораторного макету: мікроконтролер ESP32-S3, цифрові сенсори BME280 (температура, вологість, тиск) та BH1750 (освітленість), виконавчий механізм (RGB-світлодіод).
   3. Науково-технічна література: документація на компоненти (Datasheets), стандарти протоколів обміну даними, публікації щодо методів автоматизації біомедичних досліджень.

5. Перелік завдань, які потрібно розробити
   1. Розробити архітектуру програмно-апаратного комплексу, визначивши схему взаємодії компонентів системи (сенсори, мікроконтролер, брокер повідомлень, клієнтський додаток).
   2. Розробити вбудоване програмне забезпечення для мікроконтролера з використанням операційної системи реального часу FreeRTOS, реалізувавши паралельне виконання задач збору даних та підтримки мережевого з’єднання.

3. Організувати протокол обміну даними на базі стандарту MQTT, розробивши структуру топіків та формат бінарних пакетів для оптимізації швидкості передачі даних.
4. Розробити графічний інтерфейс користувача засобами мови Python та бібліотеки PyQt5, реалізувавши модульну архітектуру додатка з використанням патернів асинхронного програмування.
5. Реалізувати алгоритми обробки даних, що включають парсинг вхідних повідомлень, їх візуалізацію у вигляді графіків реального часу та збереження результатів експерименту.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу
Презентація MS Power Point

7. Орієнтовний перелік публікацій

1. Розум Є. О., Розум О. І. Розробка лабораторії на чипі для дослідження остеогенних процесів // Електроніка та нанотехнології (ELNANO-2024) : збірник праць XLIV Міжнар. конф. (Київ, 13–16 трав. 2024 р.). Київ : КПІ ім. Ігоря Сікорського, 2024.
2. Патент на корисну модель 160850 Україна, МПК C12M 3/00, G01N 15/00, G01N 29/00. Лабораторія на чипі для мікробіологічних досліджень / Ніколов М. О., Розум Є. О., Розум О. І., Карплюк Є. С.; заявник і патентовласник Нац. техн. ун-т України "Київ. політехн. ін-т імені Ігоря Сікорського". — № u202500682; заявл. 17.02.2025; опубл. 15.10.2025, Бюл. № 42.

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-5	Ніколов М. О. доц.каф.ЕІ, к.т.н., с.н.с.		

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Розробка архітектури програмно-апаратного комплексу	01.09.2025 - 20.09.2025	
2	Розробка вбудованого ПЗ для мікроконтролера (FreeRTOS)	10.09.2025 - 25.10.2025	
3	Організація протоколу обміну даними MQTT	20.09.2025 - 20.10.2025	
4	Розробка графічного інтерфейсу користувача (Python, PyQt5)	01.10.2025 - 15.11.2025	
5	Реалізація алгоритмів обробки даних	20.10.2025 - 20.11.2025	
6	Проведення експериментальних досліджень системи	10.11.2025 - 25.11.2025	
7	Оформлення магістерської дисертації	01.11.2025 - 01.12.2025	
8	Підготовка до захисту	01.12.2025 - 23.12.2025	

Студент Євген РОЗУМ

Науковий керівник Георгій СВЄЧНІКОВ

РЕФЕРАТ

Магістерська дисертація: 74 с., 13 рис., 5 табл., 43 джерела.

Магістерська дисертація присвячена розробці програмно-апаратного комплексу для автоматизованого керування та моніторингу параметрів у середовищі «Лабораторії на чипі» (ЛнЧ). Дослідження передбачає створення цілісної системи, що об’єднує мікроконтролерне керування, бездротову передачу даних та кросплатформний інтерфейс користувача.

У роботі реалізовано вбудоване програмне забезпечення на базі операційної системи реального часу FreeRTOS, що дозволило ефективно розподілити обчислювальне навантаження між ядрами мікроконтролера ESP32-S3. Для організації надійного обміну даними розроблено гібридний протокол на основі стандарту MQTT, який використовує оптимізовану бінарну серіалізацію для потокової телеметрії та JSON-формат для конфігурації пристрою. Клієнтське програмне забезпечення, створене мовою Python з використанням бібліотеки PyQt5, забезпечує асинхронну обробку даних та їх візуалізацію в режимі реального часу.

Основна мета роботи — розробка програмно-апаратного комплексу системи вимірювання біомедичних сигналів ЛнЧ. Завдяки розробленій архітектурі дослідження має на меті надати науковцям доступний та гнучкий інструмент для дистанційного контролю експериментів. Результати роботи пропонують готове до використання технічне рішення, що дозволяє значно спростити роботу з мікрофлюїдними пристроями та забезпечити високу точність реєстрації експериментальних даних.

Ключові слова: ЛАБОРАТОРІЯ НА ЧИПІ, IOT, ESP32, FREERTOS, MQTT, PYTHON, PYQT5, БІНАРНИЙ ПРОТОКОЛ, МІКРОФЛЮІДИКА, АВТОМАТИЗАЦІЯ.

ABSTRACT

Master’s thesis: 74 p., 13 fig., 5 tables., 43 sources.

The Master’s thesis is devoted to the development of a software-hardware system for automated control and parameter monitoring in a Lab-on-a-Chip (LoC) environment. The research includes the creation of an integrated system that combines microcontroller-based control, wireless data transmission, and a cross-platform user interface.

The work implements embedded software based on the FreeRTOS real-time operating system, which enables efficient distribution of computational load across the cores of the ESP32-S3 microcontroller. To ensure reliable data exchange, a hybrid communication protocol based on the MQTT standard was developed, using optimized binary serialization for streaming telemetry and the JSON format for device configuration. The client software, created in Python using the PyQt5 library, supports asynchronous data processing and real-time visualization.

The main objective of the work is to increase the efficiency of biomedical research by automating routine measurements of microclimate and illumination parameters. Thanks to the proposed architecture, the system provides researchers with an accessible and flexible tool for remote experiment control. The results of the work offer a ready-to-use technical solution that significantly simplifies interaction with microfluidic devices and ensures high accuracy in recording experimental data.

Keywords: LAB-ON-A-CHIP, IOT, ESP32, FREERTOS, MQTT, PYTHON, PYQT5, BINARY PROTOCOL, MICROFLUIDICS, AUTOMATION.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

АЦП — Аналого-цифровий перетворювач

ЛнЧ — Лабораторія на чипі

ПЗ — Програмне забезпечення

API — Application Programming Interface (Прикладний програмний інтерфейс)

CSV — Comma-Separated Values (Значення, розділені комою)

GPIO — General-Purpose Input/Output (Інтерфейс введення-виведення загального призначення)

GUI — Graphical User Interface (Графічний інтерфейс користувача)

I2C — Inter-Integrated Circuit (Послідовна міжмікросхемна шина даних)

IoT — Internet of Things (Інтернет речей)

JSON — JavaScript Object Notation (Текстовий формат обміну даними)

LED — Light-Emitting Diode (Світлодіод)

MEMS — Micro-Electro-Mechanical Systems (Мікроелектромеханічні системи)

MQTT — Message Queuing Telemetry Transport (Протокол передачі телеметрії)

PDMS — Polydimethylsiloxane (Полідиметилсилоксан)

RGB — Red, Green, Blue (Колірна модель червоний-зелений-синій)

RTOS — Real-Time Operating System (Операційна система реального часу)

SPI — Serial Peripheral Interface (Послідовний периферійний інтерфейс)

USB — Universal Serial Bus (Універсальна послідовна шина)

ВСТУП

Сучасний розвиток біотехнологій та тканинної інженерії характеризується переходом від макроскопічних досліджень до використання мікрофлюїдних систем типу «Лабораторія на чипі» (ЛнЧ). Такі системи дозволяють моделювати біологічні процеси, в умовах, максимально наближених до природних, при значному зменшенні витрат реагентів та біологічного матеріалу. Однак повноцінне використання потенціалу таких систем неможливе без надійних засобів автоматизації, що дозволяють здійснювати безперервний моніторинг та керування параметрами середовища в реальному часі. Існуючі на ринку комерційні станції моніторингу є переважно закритими пропрієтарними системами з високою вартістю, що обмежує їх доступність. Водночас розвиток технологій Інтернету речей (IoT) та поява високопродуктивних мікроконтролерів відкривають можливості для створення доступних та енергоефективних систем автоматизації. Тому розробка спеціалізованого програмного забезпечення (ПЗ), яке забезпечує надійний збір даних, дистанційне керування та візуалізацію процесів у реальному часі з використанням сучасних мережевих протоколів, є актуальним науково-прикладним завданням.

Останнім часом значна увага приділяється апаратній частині мікрофлюїдних систем, проте питання ПЗ для них часто залишається на другому плані або вирішується за допомогою дорогих пропрієтарних платформ. Існуючі дослідження підкреслюють важливість стабільності параметрів середовища для успішного культивування клітин, що вимагає переходу від ручного контролю до автоматизованих систем на базі технологій IoT.

Для реалізації поставлених задач використовуються сучасні методи розробки ПЗ, зокрема об'єктно-орієнтоване програмування, методи системного аналізу та проєктування розподілених систем. Дослідження проводиться в кілька етапів: розробка архітектури – визначення взаємодії

компонентів системи; програмна реалізація – написання коду прошивки та клієнтського додатка; налагодження протоколів – оптимізація передачі даних; експериментальне тестування – перевірка працездатності комплексу на реальному макеті з аналізом отриманих характеристик.

Метою даного дослідження є розробка програмного забезпечення користувача та вбудованого ПЗ мікроконтролера для автоматизованої системи вимірювання параметрів ЛнЧ.

Для досягнення поставленої мети необхідно вирішити такі задачі:

1. Розробити архітектуру програмно-апаратного комплексу, визначивши схему взаємодії компонентів системи (сенсори, мікроконтролер, брокер повідомлень, клієнтський додаток).
2. Розробити вбудоване програмне забезпечення для мікроконтролера з використанням операційної системи реального часу FreeRTOS, реалізувавши паралельне виконання задач збору даних та підтримки мережевого з’єднання.
3. Організувати протокол обміну даними на базі стандарту MQTT, розробивши структуру топіків та формат бінарних пакетів для оптимізації швидкості передачі даних.
4. Розробити графічний інтерфейс користувача засобами мови Python та бібліотеки PyQt5, реалізувавши модульну архітектуру додатка з використанням патернів асинхронного програмування.
5. Реалізувати алгоритми обробки даних, що включають парсинг вхідних повідомлень, їх візуалізацію у вигляді графіків реального часу та збереження результатів експерименту.

Наукова новизна даної роботи полягає у розробці програмно-апаратного комплексу системи вимірювання для мікрофдюїдних пристроїв біомедичного та наукового призначення, що в Україні створюється уперше.

Практична значимість дослідження полягає у створенні доступного та масштабованого інструменту для автоматизації рутинних лабораторних

процесів, що дозволяє підвищити точність біомедичних експериментів та забезпечити дистанційний контроль їх перебігу.

Робота складається з таких розділів: вступ – обґрунтування актуальності, визначення мети та завдань; аналіз предметної області – огляд технологій ЛнЧ та обґрунтування вибору апаратних засобів; проєктування архітектури – опис схем підключення та логіки роботи FreeRTOS; програмна реалізація – детальний опис розроблених модулів Python та алгоритмів обробки даних; експериментальні дослідження – результати тестування системи в умовах, наближених до реальних; стартап-проект – аналіз комерційного потенціалу розробки; висновки – узагальнення результатів роботи.

Розробка ПЗ для керування ЛнЧ є важливим кроком у розвитку інструментарію біомедичних досліджень. Дана робота спрямована на створення відкритої та гнучкої платформи, що може значно спростити проведення складних експериментів з дослідження остеогенезу та сприяти ширшому впровадженню мікрофлюїдних технологій у наукову практику.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОБҐРУНТУВАННЯ ВИБОРУ ЗАСОБІВ РОЗРОБКИ

У даному розділі проведено аналіз сучасного стану та перспектив розвитку технології ЛнЧ як інструменту для проведення біомедичних досліджень. Розглянуто конструктивні особливості мікрофлюїдних пристроїв та існуючі методи вимірювання фізико-хімічних параметрів у середовищі культивування клітин. На основі проведеного аналізу виконано обґрунтування вибору апаратної платформи та програмних засобів, необхідних для реалізації автоматизованої системи керування та моніторингу, а також сформульовано постановку задачі на магістерську дисертацію.

1.1 Загальна характеристика технології «Лабораторія на чипі» та сфери її застосування

Технологія ЛнЧ являє собою клас мініатюрних пристроїв, що інтегрують одну або декілька лабораторних функцій на єдиній підкладці (чипі) площею від кількох квадратних міліметрів до кількох квадратних сантиметрів. Концепція ЛнЧ базується на принципах мікрофлюїдики — науки, що вивчає поведінку рідин у мікроскопічних каналах, та технологіях мікроелектромеханічних систем (MEMS). Основною метою створення таких систем є масштабування макроскопічних хімічних та біологічних процесів до мікрорівня, що дозволяє досягти якісно нових показників ефективності досліджень.

Фундаментальною особливістю ЛнЧ є робота з надмалими об’ємами рідин, що вимірюються у нанолітрах або піколітрах. У таких масштабах течія рідини має переважно ламінарний характер, що забезпечує передбачуваність дифузійних процесів та дозволяє точно керувати перемішуванням реагентів і

транспортуванням біологічних зразків. Це створює унікальні умови для проведення клітинного аналізу, оскільки мікросередовище в каналах чіпа значно точніше імітує природні умови in vivo, ніж традиційні чашки Петрі або планшети [1, 2].

Впровадження технології ЛнЧ у наукову та медичну практику обумовлене низкою суттєвих переваг перед класичними методами лабораторної діагностики. Насамперед, це значне скорочення часу проведення аналізу завдяки високій швидкості перебігу реакцій у мікрооб'ємах та можливості паралельного виконання великої кількості тестів. Крім того, мініатюризація призводить до суттєвого зменшення витрат дорогих реагентів та зразків, що є критичним фактором при роботі з рідкісними біологічними матеріалами або при розробці нових фармацевтичних препаратів. Компактність та портативність пристроїв ЛнЧ відкривають можливості для створення систем діагностики "біля ліжка хворого" (Point-of-Care Testing), дозволяючи отримувати результати аналізів безпосередньо у місці надання медичної допомоги без необхідності транспортування зразків до централізованих лабораторій [3].

Сфери застосування ЛнЧ є надзвичайно широкими та продовжують активно розширюватися. У медицині та фармакології ці пристрої використовуються для скринінгу лікарських засобів, генетичного аналізу, детекції патогенів та біомаркерів захворювань. Окремим важливим напрямком є створення «Органів на чипі» (Organ-on-a-Chip) (рис. 1.1) — мікрофізіологічних систем, що моделюють функції людських органів для дослідження фізіології та патології тканин, а також тестування токсичності нових препаратів [4, 5].

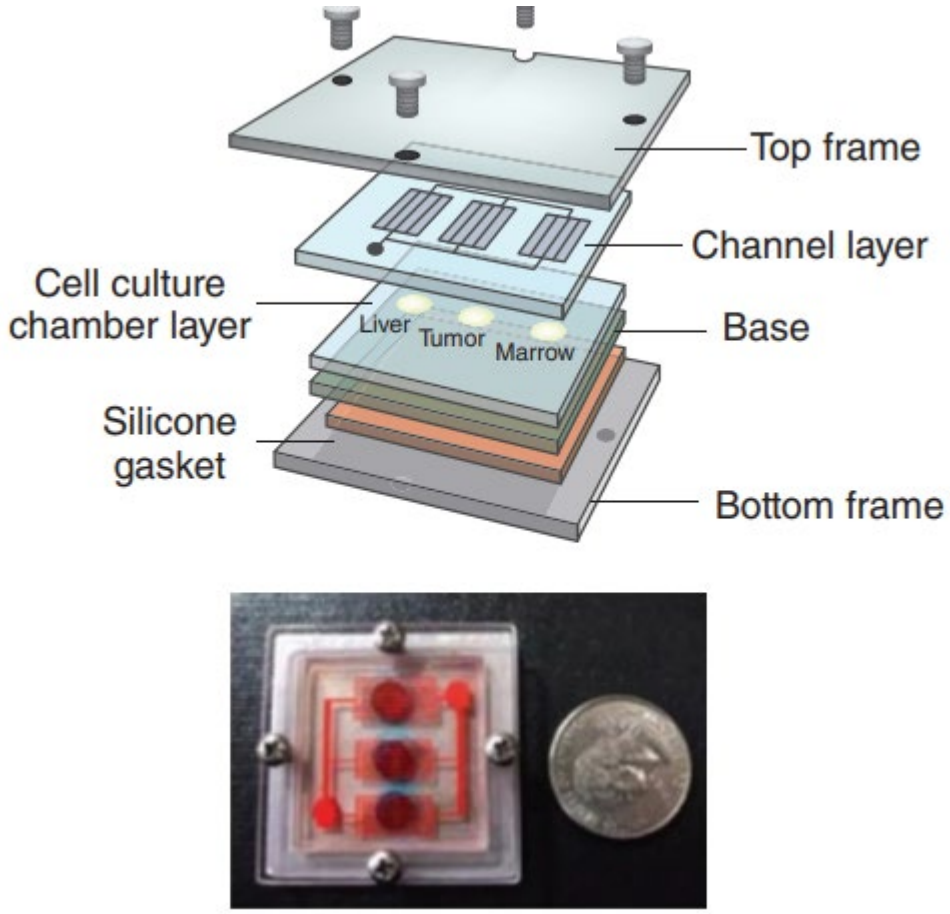

Рисунок 1.1 — Приклад реалізації концепції «Орган-на-чіпі» для моделювання біологічних процесів [4]

Однак, ефективне функціонування таких складних мікросистем неможливе без належного апаратного та програмного забезпечення. Для підтримки життєдіяльності клітинних культур у ЛнЧ необхідно забезпечити точний контроль параметрів навколишнього середовища, таких як температура, pH, концентрація газів та поживних речовин. Це зумовлює необхідність інтеграції мікрофлюїдних чіпів із сучасними сенсорними системами та засобами автоматизації, що дозволяють здійснювати моніторинг процесів у реальному часі та забезпечувати дистанційне керування експериментом. Саме розробка надійного інтерфейсу користувача та ПЗ для керування такими системами є актуальним інженерним завданням, вирішення якого дозволить повною мірою реалізувати потенціал технології ЛнЧ [6].

1.2 Класифікація та конструктивні особливості мікрофлюїдних пристроїв

Конструктивна реалізація мікрофлюїдних пристроїв базується на використанні спеціалізованих матеріалів та технологій мікрофабрикації, які визначають фізичні властивості чіпа, його вартість та сумісність із біологічними об’єктами [7]. Вибір матеріалу є критичним етапом проектування, оскільки він повинен забезпечувати не лише механічну міцність структури, але й оптичну прозорість для мікроскопії, хімічну інертність до реагентів та біосумісність, що є обов’язковою умовою для довготривалого культивування клітин.

Історично перші мікрофлюїдні пристрої виготовлялися з кремнію та скла з використанням технологій фотолітографії та травлення, запозичених з напівпровідникової промисловості. Скло характеризується відмінними оптичними властивостями, високою хімічною стійкістю та низькою автофлуоресценцією, що робить його ідеальним матеріалом для високоточних оптичних вимірювань. Однак складність обробки, крихкість та висока вартість виготовлення обмежують широке застосування скляних чіпів, особливо у випадку необхідності створення складних тривимірних структур або одноразових пристроїв.

На сучасному етапі розвитку технології домінуючу роль зайняли полімерні матеріали, серед яких найбільш розповсюдженим є полідиметилсилоксан (PDMS). Цей еластомер дозволяє створювати мікроканали методом м’якої літографії, забезпечуючи високу точність відтворення геометрії [8]. Важливою перевагою PDMS є його газопроникність, що дозволяє забезпечити необхідний газообмін для клітинних культур безпосередньо через стінки каналів [9]. Водночас, для створення більш жорстких та довговічних конструкцій широко застосовуються термопласти та епоксидні смоли. Епоксидні матеріали дозволяють формувати надійну основу

чіпа, стійку до механічних навантажень, та забезпечують герметичність з'єднань при інтеграції з іншими елементами системи.

Таблиця 1.1 — Порівняльна характеристика матеріалів для виготовлення мікрофлюїдних пристроїв [7]

Властивість	Кремній / Скло	Еластомери (PDMS)	Термореактивні пластики	Термопластики	Гідрогелі	Папір
Модуль Юнга (ГПа)	130–180 / 50–90	~0.0005	2.0 - 2.7	1.4 - 4.1	низький	0.0003-0.0025
Технологія виготовлення	Фотолітографія	Лиття (Casting)	Лиття, фотополімеризація	Термоформування	Лиття, фотополімеризація	Фотолітографія, друк
Мін. розмір каналу	<100 нм	<1 мкм	<100 нм	~100 нм	~10 мкм	~200 мкм
Профіль каналу	Обмежений 3D	3D	Довільний 3D	3D	3D	2D
Багатошаровість	Складно	Легко	Легко	Легко	Середньо	Легко
Термостійкість	Дуже висока	Середня	Висока	Середня/ Висока	Низька	Середня
Стійкість до окисників	Відмінна	Помірна	Добра	Помірна/ Добра	Низька	Низька
Сумісність з розчинниками	Дуже висока	Низька	Висока	Середня/ Висока	Низька	Середня
Гідрофобність	Гідрофільні	Гідрофобні	Гідрофобні	Гідрофобні	Гідрофільні	Амфіфільні
Поверхневий заряд	Дуже стабільний	Нестабільний	Стабільний	Стабільний	Н/Д	Н/Д
Проникність O_2 (Barrer)	<0.01	~500	0.03-1	0.05-5	>1	>1
Оптична прозорість	Ні / Висока	Висока	Висока	Середня / Висока	Низька / Середня	Низька

За принципом керування потоками рідини мікрофлюїдні пристрої класифікують на пасивні та активні системи (рис. 1.2). Пасивні пристрої використовують капілярні сили або гравітацію для транспортування рідин і не потребують зовнішніх джерел енергії, що характерно для простих діагностичних тест-смужок. Натомість складні дослідницькі платформи, зокрема системи типу «Орган-на-чіпі», відносяться до активного типу. Вони вимагають примусової подачі живильного середовища за допомогою зовнішніх шприцевих або перистальтичних насосів [10]. Перфузія дозволяє імітувати кровообіг, забезпечуючи постійне оновлення поживних речовин та видалення метаболітів, а також створює необхідне напруження зсуву рідини, що впливає на механобіологічні властивості клітин.

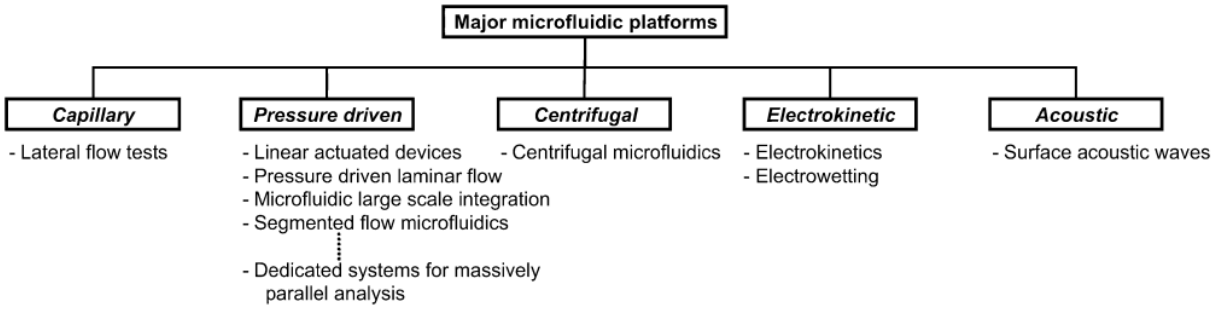

Рисунок 1.2 — Класифікація мікрофлюїдних платформ за принципом керування потоками рідини [3]

Особливу групу складають конструкції, призначені для дослідження тканинної інженерії, зокрема остеогенезу. Такі чіпи зазвичай мають складну багатошарову архітектуру, де центральний елемент — біореактор — обмежений зверху та знизу покривним склом для забезпечення можливості спостереження у реальному часі. Важливою конструктивною особливістю подібних систем є наявність спеціальних камер для розміщення тривимірних матриць або пористих вставок з біосумісних металів, наприклад, медичного титану. Це дозволяє досліджувати взаємодію клітин з поверхнею імплантатів та процеси мінералізації кісткової тканини в умовах, максимально наближених до фізіологічних [11]. Для забезпечення функціонування таких систем передбачається інтеграція мікроканалів для підведення середовища та портів для встановлення сенсорів або електродів, що дозволяє проводити комплексний моніторинг біопроцесів.

1.3 Огляд методів та систем вимірювання параметрів у «Лабораторіях на чипі»

Ефективність застосування мікрофлюїдних систем у біомедичних дослідженнях критично залежить від здатності отримувати точні кількісні дані

про стан біологічного об’єкта та параметри його мікрооточення. Традиційні методи лабораторного аналізу часто передбачають відбір проб або руйнування зразка, що унеможливлює спостереження за динамікою процесів у реальному часі. Тому сучасний розвиток систем типу ЛнЧ спрямований на інтеграцію сенсорів для неінвазивного моніторингу in situ, що дозволяє безперервно реєструвати зміни без порушення стерильності та цілісності експерименту.

Одним із найбільш поширених підходів до детекції у мікрофлюїдиці залишаються оптичні методи. Вони базуються на реєстрації змін поглинання, розсіювання або флуоресценції світла при його проходженні через досліджуване середовище. Класична мікроскопія дозволяє візуально оцінити морфологію клітин, ступінь їх розростання та формування тканинних структур. Однак використання громіздкого оптичного обладнання ускладнює автоматизацію та віддалений контроль. Альтернативою є використання інтегрованих оптоелектронних пар (світлодіод – фотодетектор) для вимірювання оптичної густини середовища (рис. 1.3). Такий підхід дозволяє опосередковано оцінювати концентрацію біомаси або детестувати зміни кольору pH-індикаторів, забезпечуючи перетворення біологічних показників у цифровий сигнал [12].

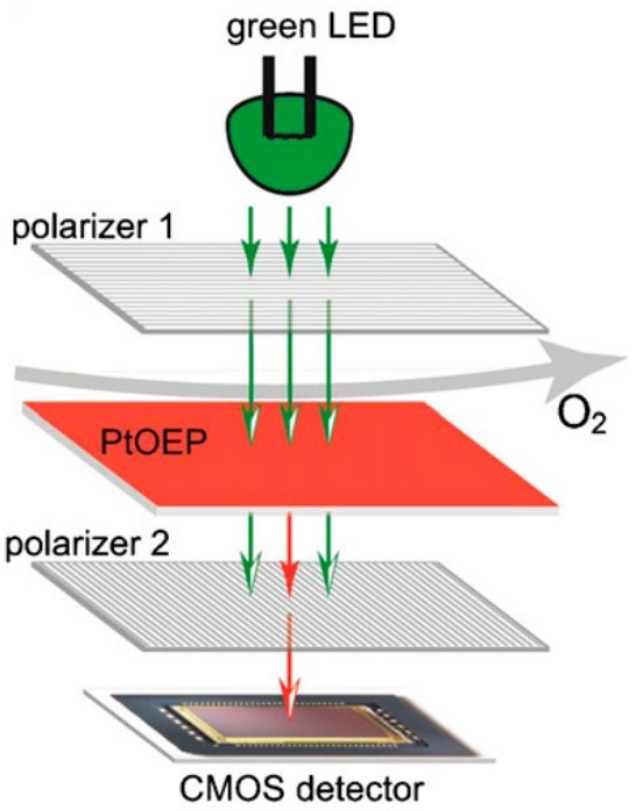

Рисунок 1.3 — Схема оптичної системи детекції параметрів середовища в мікроканалі [12]

Поряд з оптичними методами, все більшого поширення набувають електрохімічні та електрофізіологічні методи вимірювання. Вони є особливо інформативними при дослідженні збудливих тканин або процесів, пов’язаних зі зміною іонного складу середовища [13]. Для моніторингу цілісності клітинних шарів та оцінки бар’єрних функцій тканин широко застосовується метод вимірювання транс-епітеліального електричного опору або імпедансна спектроскопія. Суть методу полягає у пропусканні змінного струму слабкої потужності через систему електродів, інтегрованих у камеру чіпа (рис. 1.4). Зміна імпедансу корелює зі щільністю клітинного шару, процесами проліферації та диференціації клітин, зокрема при остеогенезі, коли відбувається мінералізація матриксу [14]. Реалізація таких вимірювань вимагає використання високоточних аналого-цифрових перетворювачів (АЦП) та спеціалізованих мікросхем для обробки біопотенціалів [15].

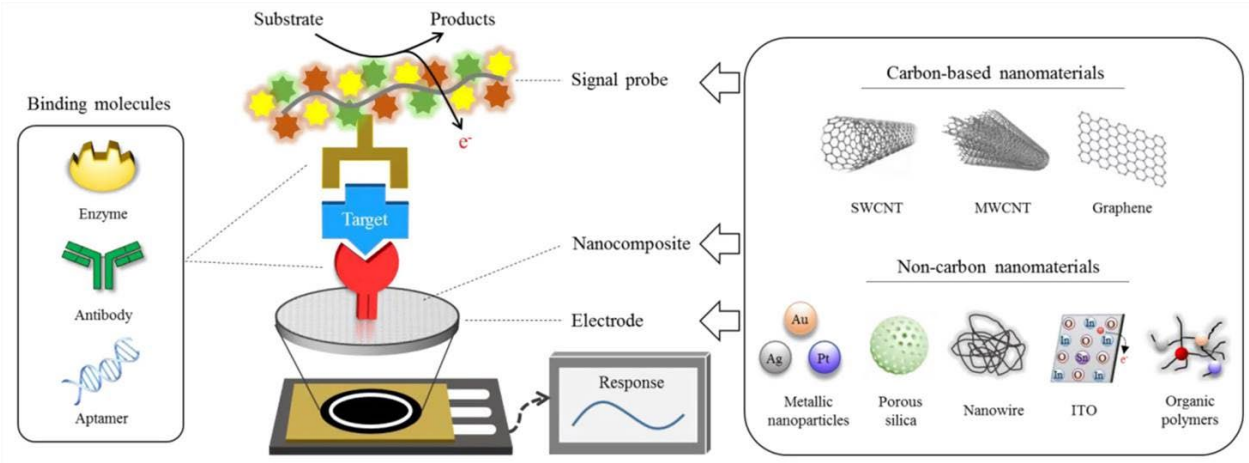

Рисунок 1.4 — Принцип роботи електрохімічного біосенсора для моніторингу клітинних процесів [13]

Крім специфічних біологічних показників, критично важливим є контроль фізичних параметрів середовища культивування, таких як температура, тиск та вологість. Температурний режим безпосередньо впливає на метаболічну активність клітин та швидкість ферментативних реакцій, тому відхилення навіть на частки градуса може спотворити результати експерименту. Контроль тиску в мікроканалах є необхідним для забезпечення стабільності потоку рідини та своєчасного виявлення аварійних ситуацій, таких як закупорка каналів або розгерметизація системи. Сучасні тенденції в розробці ЛнЧ передбачають використання мініатюрних MEMS-сенсорів для моніторингу цих параметрів, що дозволяє створювати компактні автономні платформи.

Загальною проблемою існуючих вимірювальних систем є складність одночасної обробки різнорідних даних від багатьох сенсорів та передачі їх користувачеві у зручному вигляді. Більшість комерційних рішень є закритими системами з високою вартістю, що обмежує їх гнучкість та можливість адаптації під специфічні дослідницькі задачі. Це зумовлює актуальність розробки відкритих програмно-апаратних комплексів на базі доступних мікроконтролерів та універсальних протоколів передачі даних, які здатні об’єднати керування мікрофлюїдикою та збір даних у єдину інформаційну систему.

1.4 Аналіз апаратних платформ IoT та обґрунтування вибору мікроконтролера ESP32-S3

Проектування автоматизованої системи керування для ЛнЧ висуває специфічні вимоги до апаратної обчислювальної платформи. Головний керуючий модуль повинен поєднувати в собі високу продуктивність для обробки потоків даних у реальному часі, наявність широкого спектру комунікаційних інтерфейсів для підключення периферійних сенсорів, а також інтегровані засоби бездротового зв'язку для організації віддаленого моніторингу. Крім того, важливими критеріями вибору є енергоефективність, компактність рішення та можливість реалізації багатозадачності.

На ринку вбудованих систем найбільш поширеними є рішення на базі архітектур AVR (сімейство Arduino), ARM Cortex-M (сімейство STM32) та SoC (System-on-Chip) з інтегрованими радіомодулями. Платформи на базі 8-бітних мікроконтролерів AVR є класичним рішенням для простих завдань автоматизації. Однак їх обмежена обчислювальна потужність, низька тактова частота (16 МГц) та малий обсяг оперативної пам'яті роблять їх непридатними для реалізації складних мережевих протоколів та алгоритмів шифрування. Крім того, відсутність вбудованого Wi-Fi модуля вимагає використання зовнішніх компонентів, що ускладнює схемотехніку та знижує надійність системи [16].

Більш продуктивною альтернативою є 32-бітні мікроконтролери сімейства STM32 на базі архітектури ARM Cortex-M. Вони володіють високою швидкодією, розвиненою периферією та прецизійними аналого-цифровими перетворювачами. Проте більшість бюджетних моделей цієї лінійки не мають вбудованих засобів бездротового зв'язку. Організація передачі даних по Wi-Fi або Bluetooth вимагає підключення зовнішніх трансиверів, що ускладнює програмну реалізацію драйверів та збільшує загальну вартість пристрою.

Використання ж одноплатних мікрокомп’ютерів типу Raspberry Pi є надлишковим для задач низькорівневого керування сенсорами, оскільки операційна система загального призначення (Linux) не гарантує жорсткого реального часу (Hard Real-Time), необхідного для точного зчитування біосигналів.

Оптимальним балансом між продуктивністю, функціональністю та вартістю для задач IoT є системи на кристалі серії ESP32 від компанії Espressif Systems. Для реалізації поставленої задачі обрано сучасну модифікацію мікроконтролера — ESP32-S3. Цей чіп оснащений двоядерним процесором Xtensa LX7 із тактовою частотою до 240 МГц, що дозволяє ефективно розподіляти обчислювальне навантаження [17]. Наявність двох ядер є критично важливою перевагою при використанні операційних систем реального часу, таких як FreeRTOS: одне ядро може бути виділене для підтримки стабільного Wi-Fi з’єднання та роботи стека протоколів MQTT, тоді як інше ядро безперервно обробляє дані з сенсорів без затримок, викликаних мережевою активністю.

ESP32-S3 має нативну підтримку інтерфейсів SPI та I2C, що необхідно для підключення сенсорів, а також розширені можливості керування живленням. Важливою відмінністю версії S3 від попередників є вбудований контролер USB Serial/JTAG, що спрощує налагодження та завантаження ПЗ без необхідності використання зовнішніх перетворювачів USB-UART. Великий обсяг флеш-пам’яті та оперативної пам’яті дозволяє реалізовувати буферизацію значних масивів даних перед їх відправкою на сервер, що підвищує стійкість системи до тимчасових втрат зв’язку [18]. Таким чином, сукупність технічних характеристик ESP32-S3 робить його найбільш доцільним вибором для побудови системи керування ЛнЧ.

1.5 Обґрунтування вибору інструментальних засобів розробки програмного забезпечення та протоколу MQTT

Розробка клієнтського ПЗ для керування біотехнологічними системами вимагає вибору інструментарію, який забезпечує баланс між швидкістю розробки, продуктивністю виконання та зручністю інтерфейсу користувача. В якості основної мови програмування обрано Python. Цей вибір зумовлений його статусом як де-факто стандарту в галузі наукових обчислень та обробки даних [19]. Високорівневий синтаксис та динамічна типізація Python дозволяють значно прискорити процес написання коду порівняно з компільованими мовами, такими як C++. Крім того, наявність широкого спектру спеціалізованих бібліотек для математичного аналізу та роботи з мережевими протоколами дозволяє реалізувати складну логіку програми без необхідності написання низькорівневих драйверів.

Для створення GUI обрано бібліотеку PyQt5, яка є набором прив'язок мови Python до потужного фреймворку Qt, написаного на C++. Таке поєднання дозволяє використовувати гнучкість Python для опису логіки програми, отримуючи при цьому високу продуктивність рендерингу графічних елементів, властиву нативним C++ додаткам. Важливою перевагою PyQt5 перед веб-інтерфейсами є наявність механізму сигналів та слотів, який забезпечує надійну обробку подій. Це є критичним для систем реального часу, де інтерфейс повинен миттєво реагувати на асинхронні повідомлення від датчиків, не блокуючи основний потік виконання програми. Використання інструменту Qt Designer дозволяє відділити візуальне проектування інтерфейсу від програмної логіки, що спрощує підтримку та масштабування коду.

Ключовим аспектом архітектури системи є вибір протоколу передачі даних між мікроконтролером ESP32 та комп'ютером користувача. Традиційний протокол HTTP (HyperText Transfer Protocol), що базується на архітектурі «клієнт-сервер» та моделі «запит-відповідь», є неефективним для IoT пристроїв з обмеженими ресурсами. У випадку HTTP клієнт (ПК) змушений постійно опитувати пристрій, щоб дізнатися про наявність нових даних, що створює надлишковий трафік та значні затримки. Крім того, HTTP

заголовки мають великий розмір, що є нераціональним при передачі коротких пакетів з показниками сенсорів [20].

Для вирішення цих проблем обрано протокол MQTT (Message Queuing Telemetry Transport), який працює за моделлю «публікація-підписка». У цій архітектурі пристрій та інтерфейс користувача не з’єднані безпосередньо, а обмінюються повідомленнями через проміжний вузол — брокер. Це дозволяє мікроконтролеру відправляти дані лише тоді, коли вони з’являються, і переходити в режим сну в інший час, що критично важливо для енергозбереження. MQTT оптимізований для роботи в нестабільних мережах з низькою пропускною здатністю, має мінімальний розмір заголовка пакету (2 байти) та підтримує рівні якості обслуговування (QoS), що гарантує доставку важливих повідомлень [21]. Механізм Last Will and Testament (LWT) дозволяє системі миттєво детектувати аварійне відключення пристрою, що є обов’язковою вимогою безпеки для лабораторного обладнання.

1.6 Постановка задачі на магістерську дисертацію

Проведений у першому розділі аналіз предметної області показав, що технологія ЛнЧ є перспективним інструментом для дослідження біологічних процесів, зокрема остеогенезу. Однак, існуючі комерційні рішення для автоматизації таких досліджень часто є закритими, дороговартісними та складними в адаптації під конкретні експериментальні потреби. Водночас, розвиток сучасних апаратних платформ IoT (ESP32) та програмних засобів (Python, MQTT) дозволяє створювати гнучкі та ефективні системи моніторингу власними силами.

Метою даного дослідження є розробка програмного забезпечення користувача та вбудованого ПЗ мікроконтролера для автоматизованої системи вимірювання параметрів ЛнЧ.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- Розробити архітектуру програмно-апаратного комплексу, визначивши схему взаємодії компонентів системи (сенсори, мікроконтролер, брокер повідомлень, клієнтський додаток).
- Розробити вбудоване програмне забезпечення для мікроконтролера з використанням операційної системи реального часу FreeRTOS, реалізувавши паралельне виконання задач збору даних та підтримки мережевого з’єднання.
- Організувати протокол обміну даними на базі стандарту MQTT, розробивши структуру топіків та формат бінарних пакетів для оптимізації швидкості передачі даних.
- Розробити графічний інтерфейс користувача засобами мови Python та бібліотеки PyQt5, реалізувавши модульну архітектуру додатка з використанням патернів асинхронного програмування.
- Реалізувати алгоритми обробки даних, що включають парсинг вхідних повідомлень, їх візуалізацію у вигляді графіків реального часу та збереження результатів експерименту.

2 ПРОЄКТУВАННЯ АРХІТЕКТУРИ ПРОГРАМНО-АПАРАТНОГО КОМПЛЕКСУ

У другому розділі здійснено проєктування структурної та функціональної організації системи автоматизації для ЛнЧ. Визначено вимоги до апаратної частини, розроблено загальну архітектуру ПЗ користувача та мікроконтролера. Особливу увагу приділено обґрунтуванню об’єктно-орієнтованої структури додатку, організації асинхронної мережевої взаємодії та вибору форматів передачі даних для забезпечення роботи в реальному часі.

2.1 Опис будови лабораторного макету ЛнЧ та вимог до системи керування

Об’єктом автоматизації у даній роботі виступає макет мікрофлюїдного пристрою, призначений для дослідження остеогенних процесів в контрольованих лабораторних умовах. Конструктивно пристрій виконано на основі хімічно стійкої епоксидної смоли, яка формує базовий корпус чіпа. Вибір цього матеріалу зумовлений його технологічністю, можливістю створення герметичних каналів складної геометрії та діелектричними властивостями, що є важливим при проведенні електрофізіологічних вимірювань.

Верхня та нижня площини пристрою герметизовані покривним склом. Таке рішення забезпечує оптичну прозорість конструкції, що є критичною вимогою для проведення зовнішнього спостереження та аналізу стану клітинних культур за допомогою оптичної мікроскопії без необхідності розбирання пристрою. Центральним елементом системи є камера біореактора, заповнена живильним середовищем, в якому відбуваються досліджувані біологічні процеси.

Особливістю даного макету є можливість інтеграції в робочу зону пористих вставок, виготовлених з медичного титану. Ці вставки слугують матрицею для адгезії клітин, дозволяючи моделювати процеси остеоінтеграції імплантатів. Для забезпечення життєдіяльності клітин у корпусі передбачена система мікроканалів, через які здійснюється циркуляція живильного середовища, постачання кисню та відведення продуктів метаболізму. Крім того, конструкція передбачає наявність інтегрованих електродів, які мають подвійне призначення: вони можуть використовуватися як для електричної стимуляції росту тканини, так і для зчитування біопотенціалів або вимірювання імпедансу середовища.

Виходячи з конструктивних особливостей та призначення макету, до розроблюваної системи керування та моніторингу висуваються наступні технічні вимоги: по-перше, система повинна забезпечувати безперервний моніторинг фізичних параметрів мікроклімату всередині або в безпосередній близькості до біореактора (температура, вологість, атмосферний тиск), оскільки стабільність цих показників є запорукою валідності біологічного експерименту. По-друге, необхідно реалізувати можливість зчитування специфічних електричних сигналів з електродів чіпа, що вимагає високошвидкісного інтерфейсу SPI для зв'язку зі спеціалізованими АЦП. По-третє, програмне забезпечення повинно дозволяти керувати освітленням (RGB-світлодіод) для створення необхідних умов культивування або проведення оптичних тестів.

Усі виміряні дані повинні передаватися на комп’ютер користувача в режимі реального часу з мінімальною затримкою. Враховуючи тривалість біологічних експериментів (від кількох годин до тижнів), система повинна володіти високою відмовостійкістю, автоматично відновлювати з’єднання при втраті зв’язку та забезпечувати коректну візуалізацію даних для оператора.

2.2 Характеристика апаратного інтерфейсу та конфігурація периферійних пристроїв

Апаратна реалізація системи вимірювання базується на використанні сучасного мікроконтролера ESP32-S3, який виступає центральним обчислювальним ядром комплексу. Вибір цієї платформи зумовлений необхідністю забезпечення високошвидкісної обробки даних у реальному часі паралельно з підтримкою стека бездротових протоколів. Загальна структурно-функціональна схема апаратної частини наведена на рисунку 2.1.

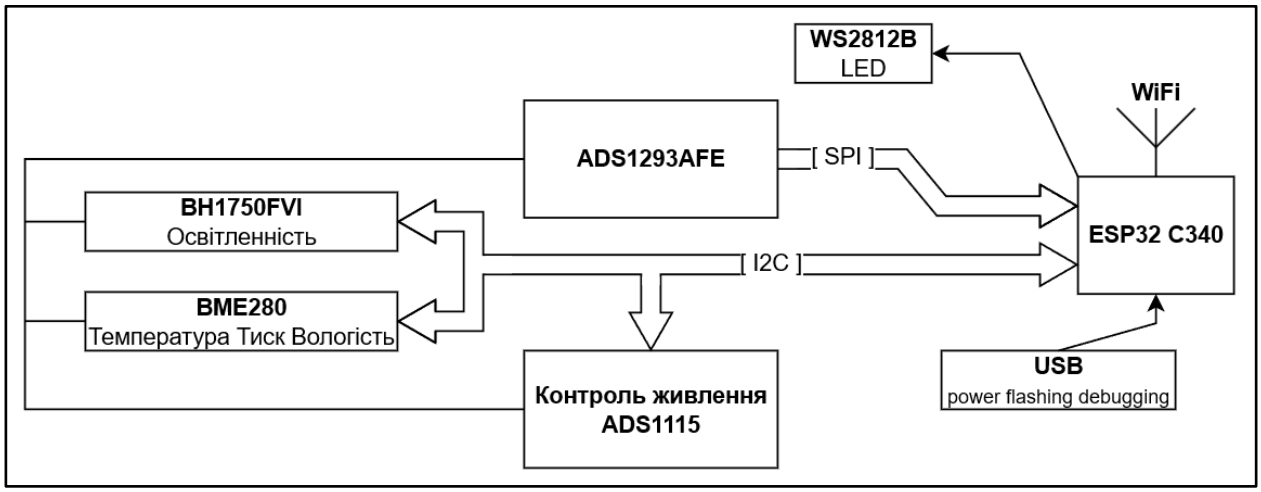

Рисунок 2.1 — Структурно-функціональна схема підключення компонентів системи

Організація обчислювального ядра та живлення працює на тактовій частоті до 240 МГц. Живлення системи здійснюється через інтерфейс USB Type-C, що забезпечує універсальність підключення. Для живлення низьковольтної логіки сенсорів використовується вбудований лінійний стабілізатор, який перетворює вхідну напругу у стабільні 3.3В. Всі логічні рівні на лініях вводу-виводу (GPIO) також відповідають стандарту 3.3В, що виключає необхідність використання додаткових перетворювачів рівнів.

Комунікаційна шина I2C Обмін даними з вимірювальними модулями організовано через послідовну шину I2C. Фізична реалізація інтерфейсу виконана на наступних виводах мікроконтролера:

- SDA (Serial Data): GPIO 8;
- SCL (Serial Clock): GPIO 9.

Програмно встановлено частоту тактування шини (I2C_FREQ) на рівні 400 кГц (Fast Mode), що забезпечує достатню пропускну здатність для опитування кількох пристроїв. У прошивці реалізовано механізм динамічного

сканування шини, який автоматично визначає адреси підключених пристроїв, забезпечуючи підтримку різних ревізій датчиків (наприклад, адреси 0x76/0x77 для BME280 та 0x23/0x5C для BH1750).

Характеристика сенсорної підсистеми

1. Сенсор мікроклімату BME280: для мінімізації похибок вимірювання, викликаних флуктуаціями повітряних потоків, сенсор налаштовано в режим MODE_NORMAL з використанням апаратного оверсемплінгу (Oversampling x2) для температури, тиску та вологості. Додатково активовано вбудований IIR-фільтр з коефіцієнтом x4, що дозволяє програмно відсіювати високочастотний шум. Встановлений час очікування (STANDBY_MS_125) запобігає самонагріву кристала, що є критичним для точності температурних вимірювань у замкнутому просторі ЛнЧ.
2. Сенсор освітленості BH1750: враховуючи необхідність реєстрації швидких змін освітленості, сенсор ініціалізується в режимі CONTINUOUS_LOW_RES_MODE. Цей режим зменшує час інтеграції фотодіода до 16 мс, що дозволяє досягти частоти дискретизації близько 60 Гц. Це забезпечує можливість детекції мерехтіння джерел світла та швидку реакцію системи керування.

До цифрового виводу GPIO 6 підключено адресний світлодіод WS2812B (NeoPixel). Керування здійснюється за однопровідним протоколом із жорсткими часовими вимогами (800 кГц). Це дозволяє використовувати світлодіод як для індикації статусів системи (наприклад, помилка сенсора, активне з'єднання MQTT), так і для керованої фотостимуляції біологічних зразків заданим спектром (RGB).

Система синхронізації реального часу Для забезпечення стабільної дискретизації сигналів, незалежної від мережевих затримок Wi-Fi, задіяно апаратний таймер мікроконтролера (hw_timer_t). Таймер сконфігуровано на частоту 1 МГц, що дозволяє генерувати переривання з точністю до мікросекунди. Це є апаратною основою для реалізації рівномірного збору даних у багатозадачному середовищі FreeRTOS.

2.3 Розробка загальної архітектури програмного забезпечення користувача

Розробка клієнтського ПЗ здійснювалася мовою програмування Python версії 3.10 із використанням кросплатформного фреймворку PyQt5. Вибір даного технологічного стека зумовлений необхідністю поєднання високої швидкості розробки, характерної для скриптових мов, із продуктивністю графічної підсистеми Qt, ядро якої реалізовано мовою C++. Загальна архітектура додатка спроєктована за модульним принципом, що дозволяє забезпечити низьку зв’язність компонентів та високу узгодженість коду. Програма розділена на три функціональні модулі, кожен з яких відповідає за окремий аспект роботи системи: збереження даних, мережеву комунікацію та взаємодію з користувачем.

Роль моделі даних у розробленій системі виконує модуль Data_General. Його основним завданням є відображення фізичних сенсорів у вигляді програмних об’єктів, що дозволяє уніфікувати роботу з різнорідними джерелами інформації. У цьому модулі реалізовано класову структуру, де кожен екземпляр сенсора виступає контейнером для збереження часових рядів вимірювань, необхідних для побудови графіків, а також окремих буферів для накопичення даних перед їх записом у файлову систему. Така організація пам’яті дозволяє чітко розмежувати оперативні дані, що використовуються для візуалізації в реальному часі ("ковзне вікно"), та архівні дані, що підлягають подальшому аналізу, забезпечуючи цілісність результатів експерименту.

За реалізацію мережевого рівня та асинхронну взаємодію з апаратною частиною відповідає модуль Mqtt_Worker. Оскільки операції вводу-виводу можуть спричиняти суттєві затримки, цей компонент винесено в окремий потік виконання за допомогою класу QThread. Це рішення дозволяє уникнути

блокування основного циклу подій графічного інтерфейсу (GUI Event Loop) під час очікування мережевих пакетів. Ключовою особливістю реалізації даного модуля є відмова від текстових форматів передачі даних на користь бінарного протоколу. Використання бібліотеки struct для десеріалізації вхідних потоків дозволяє ефективно обробляти масиви числових значень безпосередньо з байтового представлення, що суттєво знижує навантаження на процесор порівняно з парсингом JSON-структур [22]. Передача отриманих та оброблених даних до головного потоку програми здійснюється через механізм сигналів та слотів, що гарантує потокобезпечність операцій.

Функції контролера та представлення покладено на модуль Main_Class, який ініціалізує роботу системи та керує життєвим циклом додатка. Цей клас відповідає за динамічну генерацію елементів інтерфейсу на основі конфігураційних повідомлень, отриманих від мікроконтролера, що робить систему гнучкою до зміни складу підключених сенсорів без необхідності модифікації коду. Для візуалізації телеметрії інтегровано бібліотеку matplotlib, яка через спеціалізований бекенд FigureCanvasQTAgg дозволяє відображати складні наукові графіки як нативні віджети Qt. Оновлення візуальної інформації реалізовано з використанням механізму "throttling" через таймери, що дозволяє відокремити частоту надходження даних від частоти перемальовування інтерфейсу, забезпечуючи плавність роботи програми навіть при високому навантаженні [23].

2.4 Проєктування структури класів та аналіз складності коду

Для забезпечення надійності, супроводжуваності та масштабованості ПЗ на етапі проєктування було застосовано об’єктно-орієнтований підхід. Візуалізація архітектури класів та зв’язків між ними виконана за допомогою інструменту статичного аналізу Pyreverse, що входить до складу пакету Pylint, а генерація графічних схем реалізована засобами пакету Graphviz. Отримана

UML-діаграма класів (рис. 2.2) дозволяє верифікувати відповідність реалізованої структури закладеним архітектурним патернам [24].

Аналіз отриманої діаграми демонструє чітку ієрархічну організацію коду. Центральний клас системи Main_Class успадковує функціонал від базового класу QMainWindow бібліотеки PyQt5, що забезпечує інтеграцію з віконною підсистемою операційної системи. Реалізовано композиційний зв'язок з класом Data_General, який виступає агрегатором даних сенсорів. У свою чергу, Data_General керує динамічним списком екземплярів класу Sensor_class, що підтверджує реалізацію відношення «один-до-багатьох». Клас мережевої взаємодії Mqtt_Worker успадковано від QObject, що є необхідною умовою для його коректної роботи в окремому потоці та використання механізму сигналів Qt. Така структура свідчить про низьку зв’язність компонентів, оскільки логіка збереження даних та мережевий обмін інкапсульовані в окремих сутностях і не залежать від реалізації графічного інтерфейсу.

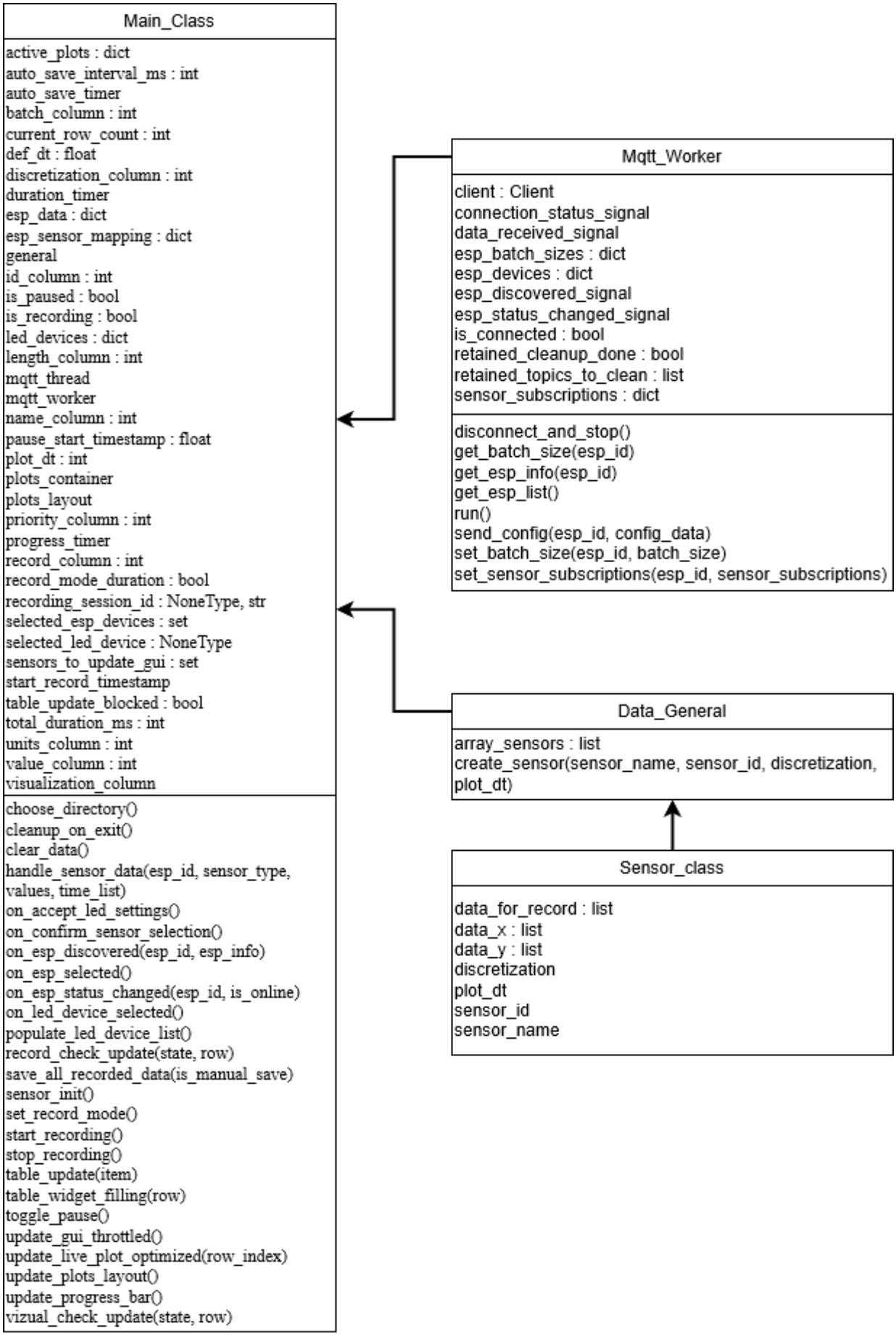

Рисунок 2.2 — UML-діаграма класів ПЗ користувача системи вимірювання ЛнЧ

З метою оцінки якості програмного коду та виявлення потенційно проблемних ділянок було проведено статичний аналіз складності за допомогою інструменту Radon (рис. 2.3). В якості основної метрики обрано цикломатичну складність Маккейба (Cyclomatic Complexity, CC), яка кількісно визначає кількість лінійно незалежних маршрутів через програмний код [25]. Цей показник безпосередньо корелює зі складністю тестування та ймовірністю виникнення помилок: чим вище значення CC, тим складніше покрити код юніт-тестами.

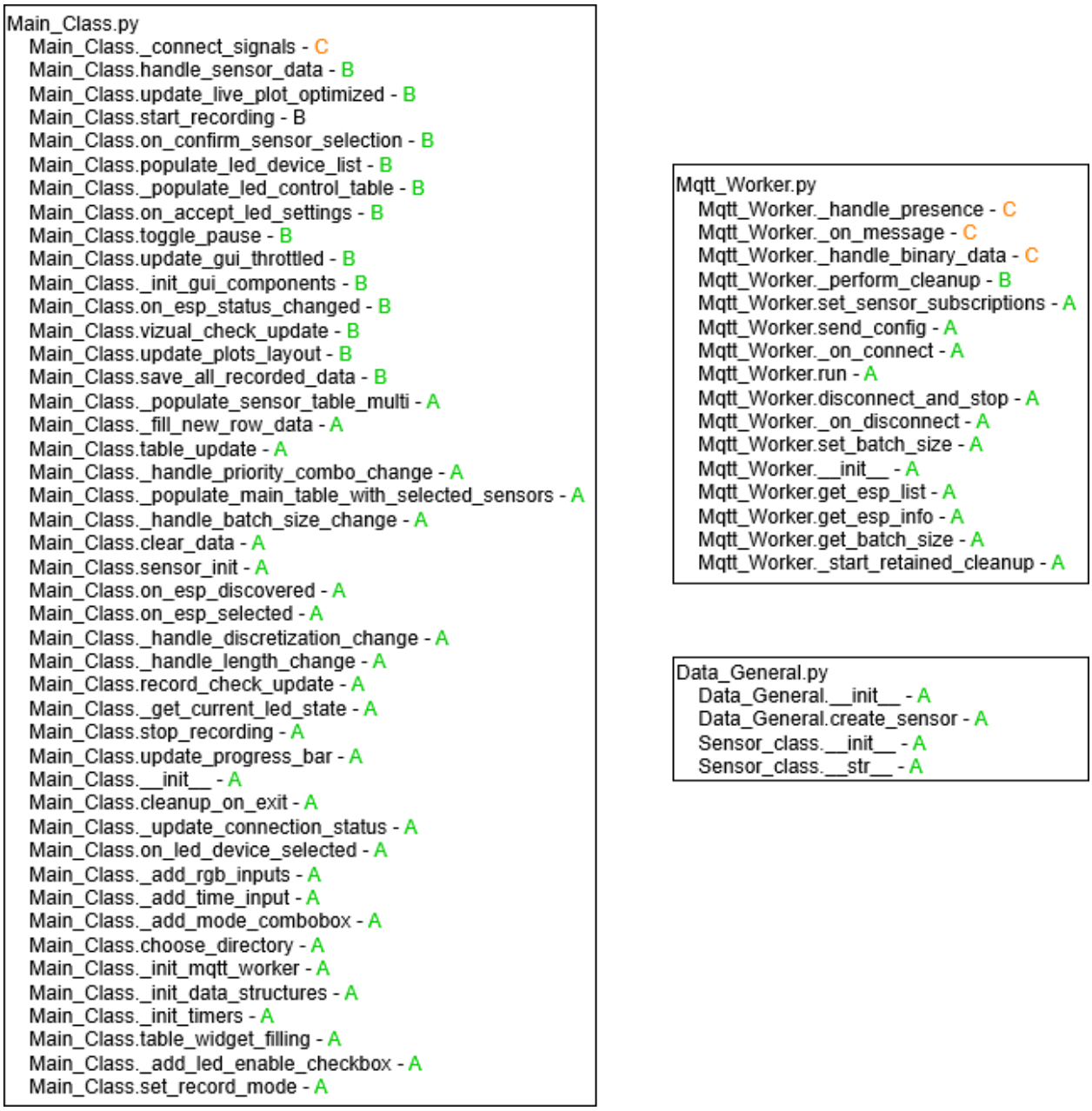

Рисунок 2.3 — Результати аналізу цикломатичної складності коду (Radon – Cyclomatic Complexity)

Проведений аудит показав, що архітектура класів загалом є збалансованою (середній показник складності класів відповідає рівню Rank A). Проте детальний аналіз виявив низку методів із підвищеною складністю (Rank B та C), що зумовлено специфікою розв'язуваних задач.

Найвищу складність (Rank C, CC 11-20) зафіксовано у методах, що містять розгалужену логіку маршрутизації та обробки виключень:

- У класі Mqtt_Worker це методи _on_message, _handle_presence та _handle_binary_data. Така оцінка є очікуваною, оскільки ці функції відповідають за парсинг вхідних даних різного формату (JSON, Binary), містять багатоступеневі перевірки структури топіків та блоки try-except для забезпечення відмовостійкості мережевого з'єднання.
- У класі Main_Class ранг C отримав метод _connect_signals. Це пояснюється необхідністю ініціалізації великої кількості зв'язків «сигнал-слот» для всіх елементів керування графічного інтерфейсу в одному місці коду.

Значна кількість методів класу Main_Class (зокрема handle_sensor_data, update_live_plot_optimized, start_recording) має середню складність (Rank B, CC 6-10). Це є характерним для додатків, де методи контролера повинні одночасно керувати станом багатьох візуальних віджетів, оновлювати графіки та взаємодіяти з даними. Натомість, модуль Data_General, що відповідає за модель даних, повністю складається з методів низької складності (Rank A), що свідчить про правильне винесення логіки збереження даних із перевантаженого інтерфейсного класу.

2.5 Організація протоколу інформаційного обміну на базі MQTT

Для забезпечення надійної та ефективної взаємодії між компонентами системи розроблено прикладний протокол обміну даними, що базується на стандарті MQTT v3.1.1 [21]. Логічна топологія мережі побудована за схемою «зірка», де центральним вузлом виступає MQTT-брокер (у даній реалізації використано публічний брокер broker.hivemq.com, порт 1883). Вибір публічного брокера дозволяє забезпечити доступ до телеметрії з будь-якої точки світу без необхідності розгортання власної серверної інфраструктури, що відповідає концепції IoT.

Адресний простір системи організовано у вигляді деревоподібної структури, що дозволяє уникнути колізій даних та забезпечити логічне групування пристроїв. Кореневий топік визначено як dezz1we/loc/main_topic. Унікальна ідентифікація кожного фізичного пристрою ЛнЧ здійснюється за його ідентифікатором (esp_id), який генерується на основі MAC-адреси мікроконтролера.

Структура топіків має наступний вигляд:

- .../{esp_id}/presence — канал виявлення пристрою (Discovery);
- .../{esp_id}/config — канал передачі налаштувань від ПК до МК;
- .../{esp_id}/{sensor_id}/data_binary — канал передачі значень вимірювань (float);
- .../{esp_id}/{sensor_id}/data_binary_time — канал передачі часових міток (uint32).

Ключовою особливістю розробленого протоколу є відмова від текстового формату JSON для передачі потокових даних. Враховуючи обмежену пропускну здатність каналу та необхідність передачі великих масивів точок для побудови графіків, застосовано бінарну серіалізацію даних. Передача здійснюється пакетами (Batch), розмір яких динамічно налаштовується (за замовчуванням 100 вимірювань).

- Формат даних: значення сенсорів передаються як масив 32-бітних чисел з плаваючою крапкою (стандарт IEEE 754, тип float), упакованих у байтовий потік з порядком байтів Little-Endian.
- Синхронізація: часові мітки передаються в окремий топік як масив 32-бітних беззнакових цілих чисел (uint32_t), що відповідають часу в мілісекундах від запуску контролера. Такий підхід дозволив зменшити об’єм корисного навантаження (payload) у 2-3 рази порівняно з JSON та виключити накладні витрати ресурсів мікроконтролера на перетворення чисел у рядки.

Для реалізації принципу Plug-and-Play розроблено механізм автоматичного виявлення пристроїв. Мікроконтролер періодично (раз на 5 секунд) публікує в топік presence повідомлення у форматі JSON, що містить:

- esp_id та mac — ідентифікатори пристрою;
- sensors — список підключених сенсорів, їх статус (enabled), пріоритет опитування та тип. Клієнтське ПЗ, отримавши це повідомлення, автоматично створює відповідні об'єкти в пам'яті та оновлює інтерфейс, не вимагаючи від користувача ручного налаштування підключення.

Передача команд від користувача до пристрою (наприклад, зміна частоти дискретизації, керування кольором RGB-світлодіода) здійснюється через топік config у форматі JSON. Використання JSON на цьому рівні є виправданим, оскільки команди передаються рідко, а читабельність та гнучкість структури даних є пріоритетними [26]. При отриманні конфігураційного пакету мікроконтролер одразу змінює параметри задач FreeRTOS або налаштування сенсорів без необхідності перезавантаження.

2.6 Проєктування логіки роботи вбудованого програмного забезпечення на базі FreeRTOS

Для забезпечення стабільності вимірювань та надійності передачі даних в умовах нестабільного бездротового зв’язку, архітектура вбудованого ПЗ мікроконтролера реалізована на базі операційної системи реального часу FreeRTOS [27]. Ключовою особливістю розробленого рішення є ефективне використання двоядерної архітектури процесора ESP32-S3 шляхом рознесення критичних до часу задач та комунікаційних процесів на різні фізичні ядра [18]. Загальна схема взаємодії задач та розподілу ресурсів наведена на рисунку 2.4.

Логіка роботи прошивки базується на паралельному виконанні двох основних задач (Tasks). Перша задача, measurementTask, відповідає за збір даних із сенсорів. Вона закріплена за ядром Core 1, яке в архітектурі ESP32 є «прикладним» і вільним від обробки системних переривань Wi-Fi стека. Для забезпечення детермінованої частоти дискретизації (1 кГц) використано механізм апаратних переривань. Апаратний таймер мікроконтролера налаштовано на генерацію переривань кожну мілісекунду. Обробник переривання (ISR) виконує мінімальну кількість інструкцій: інкрементує лічильник зразків та віддає бінарний семафор sampleSemaphore. Задача measurementTask, яка перебуває в стані блокування очікуванням цього семафора, миттєво розблоковується, зчитує показники сенсорів (BME280, BH1750, NeoPixel status) та записує їх у проміжний кільцевий буфер. Використання семафора дозволяє мінімізувати затримку вимірювань, роблячи процес збору даних незалежним від завантаженості процесора [28].

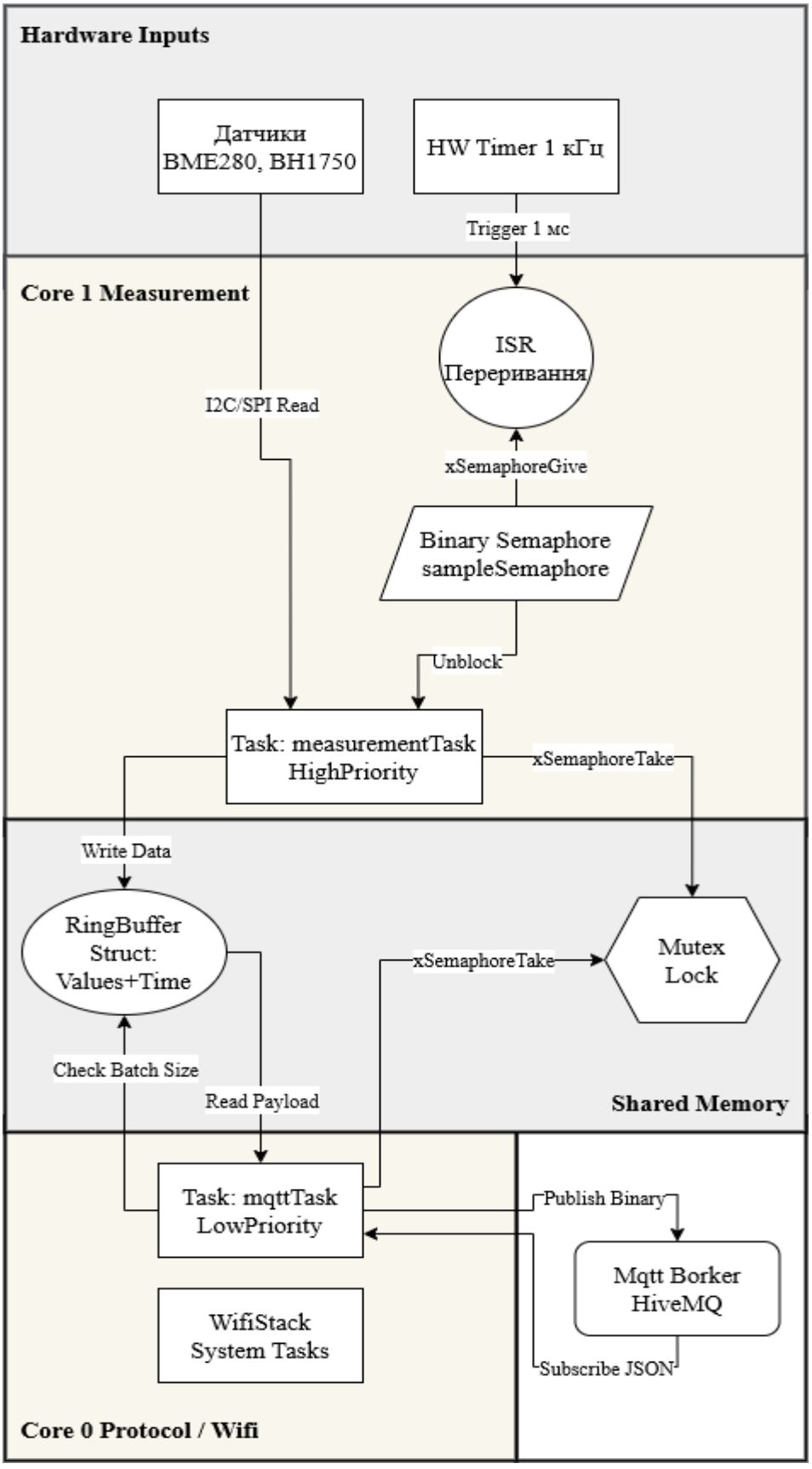

Рисунок 2.4 — Схема взаємодії задач FreeRTOS та потоків даних

Друга задача, mqttTask, відповідає за мережеву комунікацію та закріплена за ядром Core 0 («протокольним» ядром). Її пріоритет встановлено нижчим за пріоритет задачі вимірювання, що гарантує, що передача даних не перериватиме процес їх збору. У нескінченному циклі ця задача моніторить стан Wi-Fi з'єднання та підключення до MQTT-брокера, автоматично виконуючи процедуру перепідключення (Reconnect) у разі втрати зв'язку. Основною функцією задачі є перевірка наповненості кільцевих буферів сенсорів. Як тільки кількість накопичених вимірювань досягає заданого розміру пакету (Batch Size, за замовчуванням 100 елементів), задача вичитує дані з буфера та формує MQTT-повідомлення.

Для захисту цілісності даних при доступі до спільних ресурсів (кільцевих буферів) реалізовано механізм взаємного виключення за допомогою м’ютексів (xSemaphoreCreateMutex). Коли задача mqttTask вичитує дані для відправки, вона захоплює м’ютекс, блокуючи можливість запису нових даних у той самий сегмент пам'яті задачею measurementTask. Це запобігає стану гонитви (Race Condition) та пошкодженню даних [29].

Окрім потокової передачі бінарних даних, mqttTask також обробляє вхідні повідомлення конфігурації у форматі JSON. Це дозволяє динамічно змінювати параметри роботи системи, такі як пріоритети опитування сенсорів, режими роботи RGB-світлодіода (Solid, Blink, Loop) та розмір пакетів передачі, без необхідності перезавантаження пристрою. Періодично задача також публікує повідомлення присутності (presence), що містить повний зріз поточного стану системи, забезпечуючи механізм автоматичного виявлення пристрою клієнтським програмним забезпеченням.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

Третій розділ дисертаційної роботи присвячено опису практичної реалізації ПЗ автоматизованої системи вимірювання. Детально розглянуто алгоритми роботи клієнтського додатка, методи організації асинхронної мережевої взаємодії, а також програмні рішення, застосовані для обробки та візуалізації великих масивів даних у реальному часі. Окрему увагу приділено реалізації вбудованого ПЗ мікроконтролера, зокрема питанням синхронізації задач в операційній системі FreeRTOS.

3.1 Реалізація графічного інтерфейсу користувача засобами бібліотеки PyQt5

Графічний інтерфейс користувача реалізовано у класі Main_Class який є центральним модулем програми. Для прискорення процесу розробки та відокремлення візуального представлення від програмної логіки використано гібридний підхід. Базова розмітка вікна, включаючи розташування таблиць, кнопок керування та контейнерів для графіків, спроєктована у середовищі Qt Designer і збережена у форматі XML (app.ui). Завантаження цього файлу здійснюється динамічно під час ініціалізації програми за допомогою методу uic.loadUi, що дозволяє вносити зміни у дизайн без перекомпіляції коду.

Основним елементом взаємодії з оператором є головна таблиця сенсорів (MainTable), реалізована на базі віджету QTableWidget. Оскільки кількість та типи підключених сенсорів можуть змінюватися залежно від конфігурації мікроконтролера, наповнення таблиці відбувається програмно. При виборі активних сенсорів метод _fill_new_row_data створює новий рядок та ініціалізує його клітинки. Особливістю реалізації є використання складних віджетів всередині комірок таблиці. Для керування процесами запису та

відображення даних у стовпці "Record" та "Visualization" інтегровано чекбокси (QCheckBox), які для коректного центрування поміщено у проміжні контейнери QWidget із застосуванням компоновки QHBoxLayout. Для налаштування пріоритету опитування сенсорів ("Priority") динамічно створюються випадаючі списки QComboBox із варіантами частоти (100, 250, 500, 1000 Гц).

Візуалізація вимірюваних параметрів у реальному часі реалізована шляхом інтеграції бібліотеки наукової графіки matplotlib у середовище Qt. Для цього використано спеціалізований бекенд FigureCanvasQTAgg [23]. Кожен графік є окремим об'єктом FigureCanvas, який програмно додається у сітку розміщення QGridLayout всередині прокручуваної області (QScrollArea). Логіка розміщення графіків автоматично адаптує інтерфейс: при активації чекбоксу візуалізації викликається метод vizual_check_update, який створює нове полотно, налаштовує осі та легенду. У випадку RGB-сенсора створюється три окремі лінії (червона, зелена, синя) на одному графіку для відображення спектральних компонент. При деактивації чекбоксу відповідний віджет видаляється з пам'яті та інтерфейсу, а сітка графіків перебудовується для оптимального використання екранного простору.

Окремою частиною інтерфейсу є панель керування виконавчими механізмами, зокрема RGB-світлодіодом. Для цього розроблено метод _populate_led_control_table, який генерує елементи керування у таблиці SensorTable_Param. Для точного налаштування параметрів кольору використано віджети QSpinBox із валідацією діапазону 0–255 та візуальним фарбуванням фону у відповідний колір (червоний, зелений, синій). Вибір режиму роботи (статичне світло, мигання, циклічна зміна кольорів) реалізовано через QComboBox. Зчитування параметрів з усіх полів вводу та формування конфігураційного пакету здійснюється методом on_accept_led_settings, який готує дані для відправки через MQTT.

Важливим аспектом реалізації GUI є забезпечення чуйності інтерфейсу при високій частоті надходження даних. Пряме оновлення віджетів при отриманні кожного пакету призвело б до "заморожування" програми. Для

вирішення цієї проблеми застосовано патерн "Throttling". Оновлення числових значень у таблиці та перемальовування ліній на графіках виконується методом update_gui_throttled, який викликається таймером QTimer із фіксованим інтервалом 100 мс. Це дозволяє накопичувати вхідні дані у буферах, візуалізуючи лише актуальний зріз, що значно знижує навантаження на графічну підсистему [30].

3.2 Розробка модуля асинхронної мережевої взаємодії Mqtt_Worker

Модуль Mqtt_Worker є ключовим компонентом архітектури, що відповідає за стабільність та безперервність обміну даними між комп’ютером та мікроконтролером. Оскільки мережеві операції вводу-виводу є потенційно блокуючими (наприклад, очікування відповіді від сервера при поганому з'єднанні), їх виконання в основному потоці програми призвело б до "зависання" графічного інтерфейсу. Для вирішення цієї проблеми клас Mqtt_Worker спроєктовано як незалежний об'єкт, що виконується в окремому потоці (QThread).

Клас Mqtt_Worker ініціалізує клієнт MQTT (бібліотека paho-mqtt) з підтримкою версії протоколу v3.1.1. У методі run, який є точкою входу потоку, встановлюється з'єднання з брокером (broker.hivemq.com) та запускається нескінченний цикл обробки подій client.loop_forever(). Цей цикл автоматично керує підтримкою з'єднанняь та відновленням сесії при розриві зв'язку [21]. Статус підключення транслюється в основну програму через сигнал connection_status_signal, що дозволяє візуально відображати стан мережі в GUI.

Обробка вхідних даних реалізована у методі зворотного виклику _on_message. Отримавши повідомлення, алгоритм аналізує його топік, розділяючи рядок за символом “/”. Це дозволяє динамічно визначати тип повідомлення та джерело даних. Для автоматичного виявлення пристроїв

реалізовано метод _handle_presence. При отриманні JSON-пакету в топіку presence, worker перевіряє, чи зареєстровано цей esp_id у локальному реєстрі self.esp_devices. Якщо пристрій новий, він додається до списку, а його конфігурація (список сенсорів, MAC-адреса) передається у головне вікно через сигнал esp_discovered_signal. Також реалізовано механізм очищення "застарілих" retained-повідомлень (_start_retained_cleanup), щоб уникнути дублювання неактивних пристроїв при запуску програми.

Декодування бінарного потоку даних Найбільш критичною частиною модуля є метод _handle_binary_data, який відповідає за швидку обробку телеметрії. Враховуючи, що мікроконтролер відправляє дані пакетами (Batch), метод реалізує логіку буферизації та синхронізації. Дані надходять у два різні топіки:

1. data_binary — масив значень (float, 4 байти);
2. data_binary_time — масив часових міток (uint32, 4 байти).

Для їх розбору використано модуль struct. Рядок формату формується динамічно на основі розміру пакету: <{batch_size}f для значень та <{batch_size}L для часу, де символ < вказує на порядок байтів Little-Endian (стандарт для ESP32) [22]. Метод перевіряє цілісність пакету, порівнюючи довжину payload з очікуваною (кількість байт = batch_size × 4). Лише після успішного отримання та розпакування обох компонентів (часу та значень) сформований масив передається в інтерфейс через сигнал data_received_signal. Такий підхід гарантує, що на графіку кожне значення буде точно прив'язане до свого часу вимірювання.

Керування конфігурацією Модуль також забезпечує зворотний зв'язок, дозволяючи відправляти команди на пристрій через метод send_config. Перед відправкою дані (наприклад, нові параметри RGB-світлодіода або зміна розміру пакету batch_size) серіалізуються у JSON та публікуються у топік .../config. Це дозволяє змінювати поведінку мікроконтролера одразу без переривання експерименту.

3.3 Алгоритмічне забезпечення обробки даних та візуалізації в реальному часі

Ефективність ПЗ системи моніторингу визначається його здатністю обробляти високочастотні потоки даних без втрат інформації та затримок візуалізації. Для забезпечення стабільної роботи додатку було розроблено комплекс алгоритмів, що охоплюють процеси організації пам'яті, математичної обробки часових рядів та оптимізованого рендерингу графічної інформації.

Центральним елементом підсистеми обробки даних виступає об'єктна модель, реалізована у класі Sensor_class. З метою оптимізації використання оперативної пам'яті та забезпечення швидкодії застосовано стратегію роздільного буферизації. Для кожної вимірюваної величини створюються два незалежні типи сховищ: оперативні буфери, що містять дані лише за останній проміжок часу, визначений налаштуваннями вікна відображення, та буфери накопичення, які зберігають повну історію вимірювань до моменту їх запису у файл. Така архітектура дозволяє відокремити процес візуалізації, який працює з обмеженим масивом точок, від процесу логування, що оперує значними обсягами інформації, запобігаючи переповненню пам'яті при тривалих експериментах.

Алгоритм первинної обробки вхідного потоку ініціюється при отриманні пакету даних від мережевого модуля. Першим етапом є синхронізація часової шкали. Оскільки мікроконтролер передає часові мітки у форматі відносного часу роботи пристрою (uptime), програмне забезпечення виконує їх нормалізацію, перераховуючи у секунди відносно моменту початку експерименту або підключення сенсора. На наступному етапі реалізовано механізм «ковзного вікна». Нові значення додаються в кінець оперативних буферів, після чого алгоритм перевіряє сумарну кількість точок або часове охоплення. При перевищенні заданого ліміту найстаріші дані автоматично

видаляються з початку масиву, що забезпечує стабільність споживання ресурсів процесора незалежно від часу роботи системи [31].

Для відображення даних реалізовано алгоритм оптимізованої візуалізації, спрямований на мінімізацію навантаження на графічне ядро. Замість повного перемальовування полотна графіка при кожному оновленні, що є ресурсоємною операцією, використовується метод оновлення координат об'єктів ліній. Це дозволяє змінювати лише геометрію кривої без перерахунку фонових елементів сітки та осей. Для адаптації відображення під динамічний діапазон сигналів застосовано алгоритм автомасштабування осей, який обчислює екстремуми у поточному вікні даних та коригує межі відображення з додаванням відступів для покращення читабельності. Специфічний підхід застосовано для обробки даних RGB-сенсора, де одне цілочисельне значення кольору декомпозується на три складові (червону, зелену та синю) за допомогою побітових операцій зсуву та маскування, після чого будуються три синхронізовані графіки.

Збереження результатів експерименту забезпечується підсистемою логування, побудованою на базі бібліотеки pandas. Алгоритм запису працює в асинхронному режимі, періодично вивантажуючи накопичені дані з буферів у CSV-файли. Використання режиму дописування (append) дозволяє поступово формувати файл звіту, знижуючи ризик втрати даних у разі аварійного завершення роботи програми та зменшуючи пікове навантаження на дискову підсистему комп'ютера [32].

3.4 Програмна реалізація задач та черг у середовищі FreeRTOS для ESP32

Реалізація вбудованого ПЗ мікроконтролера базується на використанні можливостей операційної системи реального часу FreeRTOS, що дозволило ефективно розподілити обчислювальне навантаження між двома ядрами

процесора ESP32-S3. Логіка роботи системи побудована на взаємодії двох незалежних потоків виконання (Tasks): задачі високоточного вимірювання та задачі мережевої комунікації [27].

Ключовим елементом системи збору даних є задача measurementTask, яка відповідає за опитування сенсорів із суворо заданою періодичністю. Для забезпечення детермінованості часових інтервалів, незалежної від виконання іншого коду, використано механізм апаратних переривань. Таймер мікроконтролера налаштовано на генерацію переривань з частотою 1 кГц. Обробник переривання виконує мінімальний набір інструкцій, віддаючи бінарний семафор sampleSemaphore, який слугує тригером для розблокування основної задачі вимірювання. Такий підхід дозволяє мінімізувати час перебування у перериванні та гарантує стабільність частоти дискретизації навіть при пікових навантаженнях на процесор.

Отримавши управління після розблокування семафором, задача measurementTask виконує зчитування фізичних величин із підключених сенсорів. Для збереження результатів розроблено структуру даних типу «Кільцевий буфер» (Ring Buffer). Кожен активний сенсор має власний виділений буфер у динамічній пам'яті, куди записуються значення вимірювань та відповідні часові мітки. Використання кільцевої структури дозволяє організувати безперервний запис даних за принципом FIFO (First In, First Out) без необхідності постійного виділення та звільнення пам'яті, що є критично важливим для запобігання фрагментації кучі при тривалій роботі пристрою.

Паралельно з процесом вимірювання функціонує задача mqttTask, яка відповідає за передачу даних на сервер. Ця задача має нижчий пріоритет та виконується на іншому ядрі процесора, що ізолює процес вимірювання від можливих затримок, пов'язаних з роботою стека Wi-Fi. Алгоритм роботи задачі полягає у моніторингу наповненості кільцевих буферів сенсорів. Як тільки кількість нових вимірювань досягає встановленого розміру пакету (Batch Size), задача ініціює процедуру читання.

Для забезпечення цілісності даних при одночасному доступі до буфера з двох різних задач (вимірювання — запис, комунікація — читання)

реалізовано механізм взаємного виключення за допомогою м'ютексів (Mutex). Перед початком читання даних задача mqttTask захоплює м'ютекс, тимчасово блокуючи можливість запису в буфер. Після копіювання блоку даних у локальний масив для відправки м'ютекс звільняється. Це запобігає виникненню стану гонитви та гарантує, що дані не будуть пошкоджені або частково перезаписані під час формування мережевого пакету [28]. Сформовані бінарні пакети відправляються через MQTT-клієнт, забезпечуючи високу пропускну здатність каналу зв'язку.

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ТА ТЕСТУВАННЯ

Четвертий розділ дисертаційної роботи присвячено комплексній перевірці працездатності розробленої автоматизованої системи. Основна увага приділена верифікації проектних рішень, прийнятих на етапах розробки архітектури та програмної реалізації. У розділі наведено методику проведення випробувань, результати тестування стабільності мережевого обміну даними, оцінку точності вимірювань сенсорної підсистеми, а також аналіз реакції системи керування на динамічні зміни параметрів об'єкта дослідження.

4.1 Методика проведення випробувань розробленого комплексу

Метою проведення експериментальних досліджень є підтвердження відповідності технічних характеристик розробленого програмно-апаратного комплексу вимогам, сформульованим у першому розділі роботи. Випробування проводилися на базі виготовленого лабораторного макету ЛнЧ (рис. 4.1), оснащеного мікроконтролером ESP32-S3 та комплектом сенсорів (BME280, BH1750), зовнішній вигляд якого наведено на рисунку 4.1. В якості керуючого терміналу використовувався персональний комп'ютер під управлінням операційної системи Windows 11 зі встановленим середовищем виконання Python 3.10 та розробленим клієнтським додатком. Мережева інфраструктура була організована за допомогою бездротової точки доступу стандарту Wi-Fi 4 (802.11n), що працює на частоті 2.4 ГГц, а обмін повідомленнями здійснювався через публічний MQTT-брокер HiveMQ [33].

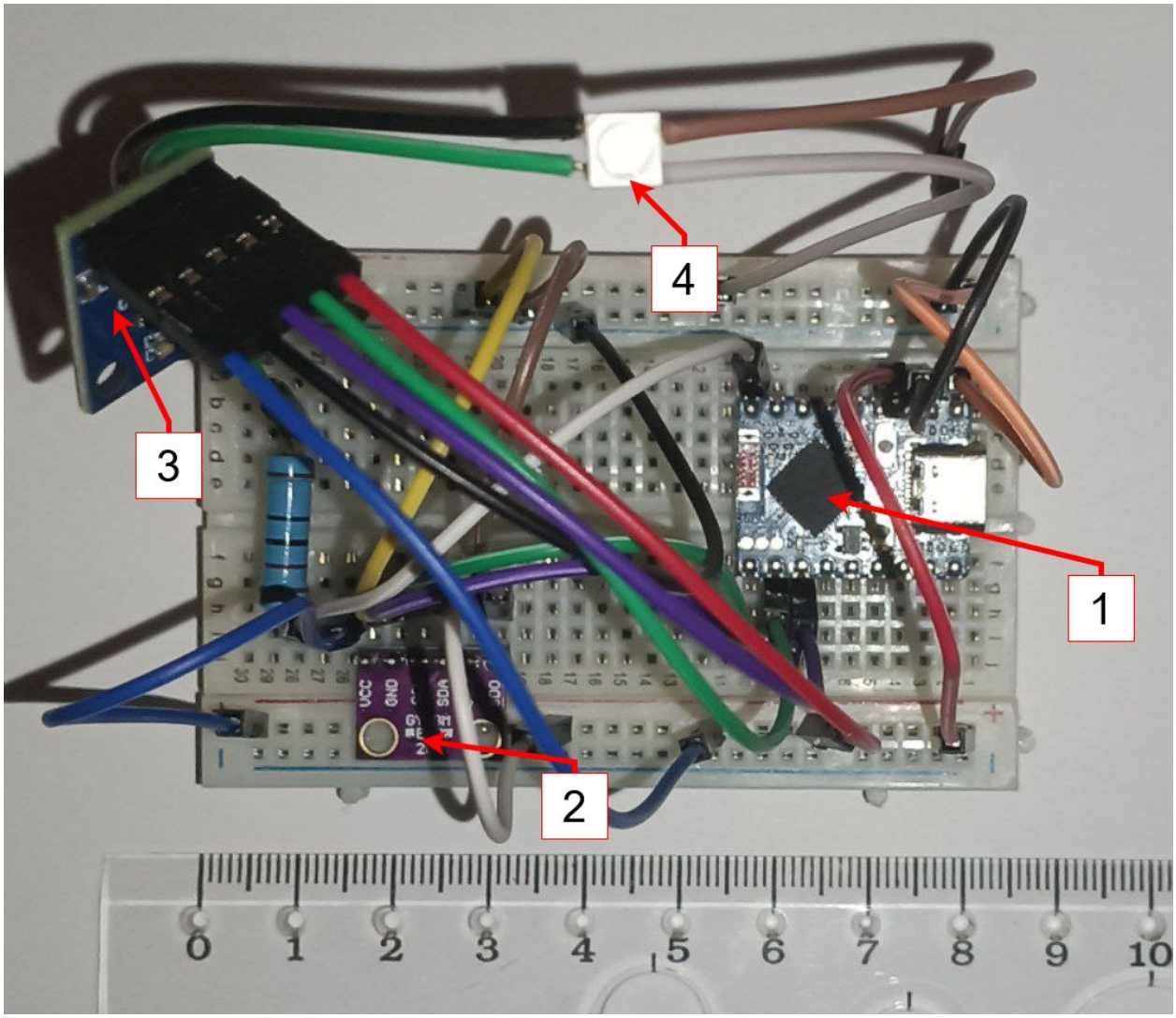

Рисунок 4.1 — Стенд вимірювальної системи ЛнЧ (спільна розробка зі студентом А. Кулаковим, гр. ДМ-41мн; особистий внесок автора — програмування мікроконтролера): 1 — мікроконтролер ESP32; 2 — датчик параметрів мікроклімату BME280; 3 — датчик освітленості BH1750; 4 — світлодіод WS2812B

Методика перевірки функціональності ПЗ включала етап статичного налагодження та етап динамічного навантажувального тестування. Для верифікації коректності роботи комунікаційних протоколів використовувалося спеціалізоване програмне забезпечення MQTT Explorer, яке дозволяло виконувати моніторинг трафіку в реальному часі [34]. Це дало можливість контролювати структуру топіків, коректність формування JSON-пакетів конфігурації та цілісність бінарних пакетів телеметрії. Перевірка алгоритмів десеріалізації даних на стороні клієнта здійснювалася шляхом порівняння вихідних значень, згенерованих сенсорами, із даними,

відображеними у графічному інтерфейсі та записаними у CSV-файли, що дозволило виключити помилки при конвертації типів даних та розрахунку часових міток.

Випробування сенсорної підсистеми проводилися шляхом створення контрольованих збурень параметрів навколишнього середовища. Для тестування каналу вимірювання температури та вологості (BME280) застосовувався метод локального нагріву корпусу пристрою з подальшим природним охолодженням, що дозволило оцінити інерційність сенсора та швидкість реакції графічного інтерфейсу на зміни. Перевірка каналу вимірювання освітленості (BH1750) здійснювалася за допомогою джерела світла зі змінною інтенсивністю та спектральним складом, при цьому фіксувалася здатність системи коректно відображати різкі перепади рівня інсоляції при високій частоті дискретизації [35].

Окремим етапом досліджень стала перевірка надійності системи керування виконавчими механізмами та стабільності роботи в умовах довготривалого експерименту. Тестування зворотного зв'язку виконувалося шляхом відправки команд керування RGB-світлодіодом із різними параметрами кольору та режимів мигання, з одночасним візуальним контролем реакції макету та моніторингом відповідних статусних повідомлень у системі. Стійкість до мережевих збоїв перевірялася шляхом примусового розриву з'єднання з точкою доступу та аналізу процесу автоматичного відновлення зв'язку, повторної підписки на топіки та відновлення потоку даних без необхідності перезавантаження клієнтського додатку.

4.2 Тестування стабільності передачі даних та швидкодії інтерфейсу

Перевірка стабільності інформаційного обміну проводилася шляхом моделювання аварійних ситуацій у мережевій інфраструктурі. Одним із ключових сценаріїв тестування була імітація тимчасової втрати з'єднання з бездротовою точкою доступу. Під час експерименту живлення маршрутизатора вимикалося на час до 60 секунд. Результати моніторингу показали, що програмне забезпечення мікроконтролера, завдяки реалізації алгоритму reconnectMQTT у задачі mqttTask, успішно детектувало розрив з'єднання та переходило в режим очікування [21]. Після відновлення мережі пристрій автоматично встановлював зв'язок з брокером та відновлював передачу телеметрії без необхідності фізичного перезавантаження. На стороні клієнтського додатка клас Mqtt_Worker коректно обробляв подію відключення, генеруючи відповідний сигнал для візуальної індикації статусу "Offline" в інтерфейсі користувача, та автоматично відновлював підписку на топіки після відновлення з'єднання.

Окрему увагу було приділено аналізу ефективності розробленого бінарного протоколу передачі даних порівняно з традиційним форматом JSON. Для цього було проведено вимірювання обсягу трафіку при передачі пакету з 100 вимірювань (Batch Size = 100). При використанні текстового формату JSON (з ключами та роздільниками) розмір повідомлення становив би приблизно 3-4 КБ. Натомість, застосування бінарної упаковки (struct) дозволило скоротити розмір корисного навантаження до 400 байт для значень (100 чисел типу float по 4 байти) та 400 байт для часових міток. Таке зменшення обсягу даних майже на порядок суттєво знизило навантаження на мережевий канал та зменшило затримки передачі (latency), що підтверджується плавністю відображення графіків у реальному часі.

Оцінка швидкодії графічного інтерфейсу користувача проводилася під час стрес-тестування, коли на вхід системи подавався потік даних із максимальною частотою оновлення. Завдяки архітектурному рішенню з винесенням мережевого клієнта в окремий потік QThread та використанню таймера для обмеження частоти перемальовування віджетів (метод update_gui_throttled), інтерфейс залишався чуйним до дій користувача.

Додатково було протестовано функцію динамічної зміни розміру пакету (batch_size) через відправку конфігураційних команд. Експериментально встановлено, що оптимальним значенням для даної конфігурації мережі є пакет розміром 50-200 вимірювань. Зменшення розміру пакету нижче 50 призводило до збільшення накладних витрат протоколу TCP/IP, тоді як збільшення понад 200 викликало помітну візуальну затримку оновлення графіків через час, необхідний для накопичення буфера на стороні мікроконтролера. Зовнішній вигляд головного вікна програми під час прийому та візуалізації даних наведено на рисунку 4.2.

Рисунок 4.2 — Головне вікно програми в режимі моніторингу даних у реальному часі: 1 — Область динамічної візуалізації даних (відображення температури, вологості, тиску, освітленості та RGB-статусу); 2 — Таблиця конфігурації та моніторингу сенсорів (відображення поточних значень, налаштування частоти опитування та вибір каналів для запису/відображення); 3 — панель керування запису експеримента (вибір режиму запису, старт/пауза/стоп, індикація прогресу); 4 — налаштування збереження файлів (вибір директорії для збереження CSV-файлів та активація режиму запису).

4.3 Перевірка точності збору даних із сенсорів в умовах, наближених до реальних

Валідація вимірювальних каналів системи здійснювалася шляхом проведення серії експериментів, спрямованих на оцінку динамічних характеристик сенсорів та коректності візуалізації даних програмним забезпеченням. Перший етап випробувань стосувався підсистеми контролю мікроклімату, реалізованої на базі сенсора BME280. Для перевірки реакції системи на зміну температури було змодельовано процес нагріву корпусу макету зовнішнім джерелом тепла з подальшим природним охолодженням до кімнатної температури. Отримані графічні залежності (рис. 4.3), демонструють плавну зміну показників без розривів та аномальних сплесків. Це підтверджує ефективність обраної частоти дискретизації та коректність роботи алгоритмів буферизації даних у класі Sensor_class. Програмне забезпечення успішно реєструвало навіть незначні флуктуації температури (в межах 0.1°C), що є критично важливим для підтримки стабільного середовища в біореакторі [36].

Рисунок 4.3 — Графік зміни температури та вологості у часі при тестовому нагріві

Наступним етапом стала перевірка оптичної підсистеми на базі сенсора BH1750. Оскільки у прошивці мікроконтролера цей сенсор налаштовано на режим низької роздільної здатності для забезпечення високої частоти оновлення (близько 60 Гц), метою тесту було підтвердження здатності ПЗ обробляти швидкі зміни світлового потоку. Експеримент полягав у створенні серії світлових імпульсів різної тривалості та інтенсивності. Аналіз отриманих графіків (рис. 4.4) показує, що система швидко реагує на зміни освітленості, а візуалізація відбувається з мінімальною затримкою. Використання бінарного протоколу передачі дозволило передавати щільний потік даних без втрат пакетів, що дозволяє використовувати дану систему не лише для моніторингу загального рівня освітлення, але й для реєстрації швидкоплинних оптичних явищ [37].

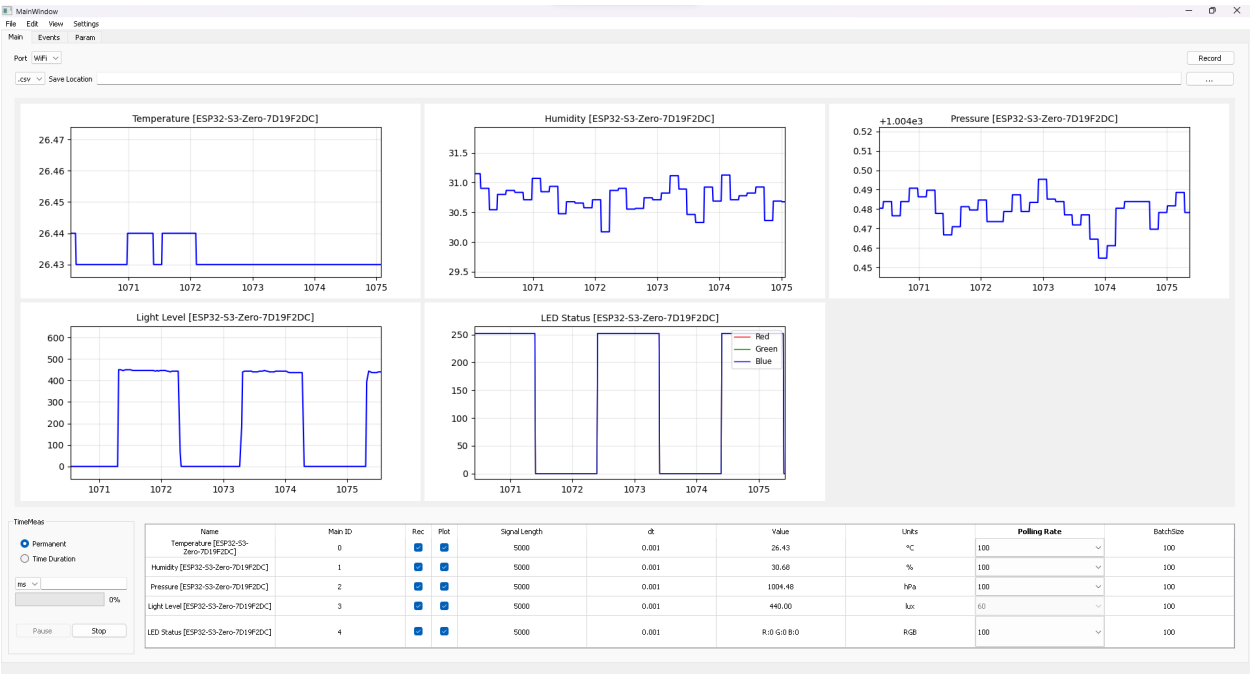

Рисунок 4.4 — Реакція системи на динамічну зміну освітленості (дані з сенсора BH1750 Light Level)

Окрему увагу було приділено перевірці точності відображення статусу виконавчих механізмів, зокрема RGB-світлодіода. У розробленому протоколі поточний колір світлодіода передається як віртуальний сенсор (SENSOR_LED_STATUS), значення якого є 24-бітним цілим числом. На стороні клієнтського додатка реалізовано алгоритм декомпозиції цього числа

на окремі колірні канали. Під час тестування оператор послідовно змінював колір світіння через панель керування (червоний -> зелений -> синій). На графіку (рис. 4.5) чітко прослідковується кореляція між відправленими командами та отриманою телеметрією: при активації червоного кольору відповідна лінія на графіку піднімалася до рівня 255, тоді як інші залишалися на нулі. Цей тест підтверджує надійність двостороннього зв'язку: команда успішно доходить до пристрою, обробляється, змінює фізичний стан світлодіода, і цей новий стан коректно зчитується та повертається у графічний інтерфейс [38].

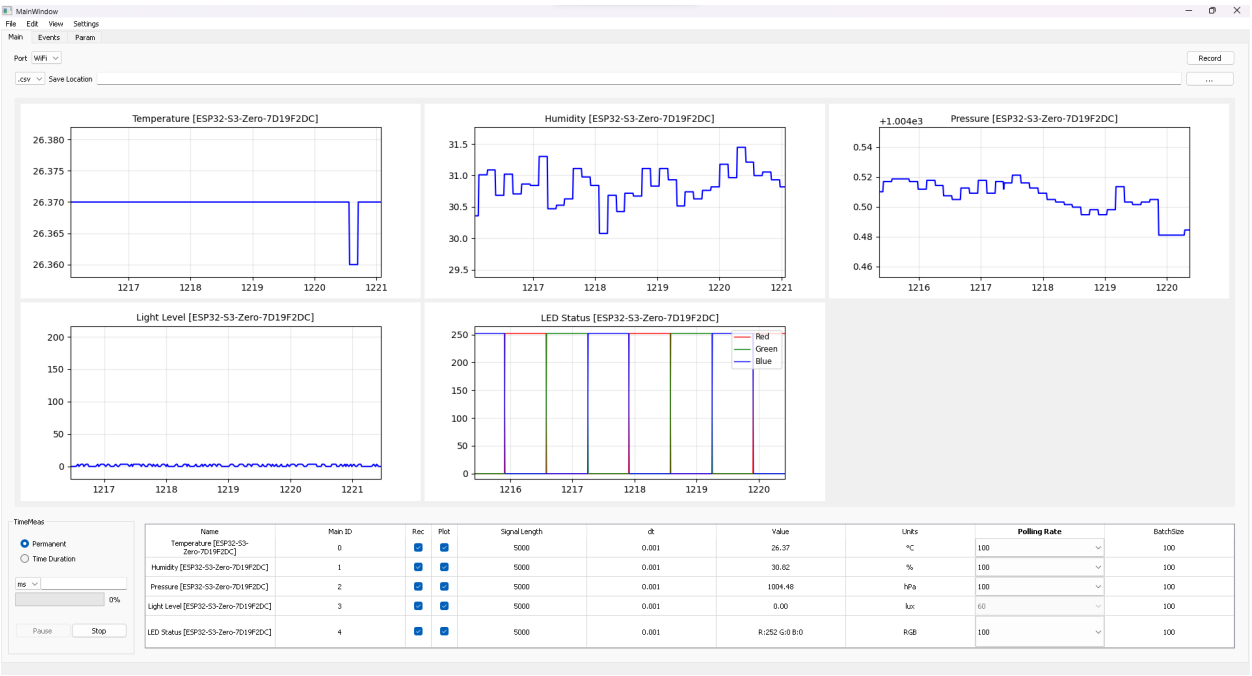

Рисунок 4.5 — Візуалізація спектральних компонент (RGB) стану світлодіода у реальному часі

Результати експериментальних досліджень підтверджують, що розроблений програмно-апаратний комплекс забезпечує необхідну точність та швидкодію для використання у лабораторних умовах. Алгоритми обробки даних коректно інтерпретують вхідні бінарні потоки, а графічний інтерфейс забезпечує інформативну візуалізацію фізичних процесів у реальному часі.

4.4 Аналіз отриманих результатів та оцінка ефективності системи

Узагальнюючи результати проведених експериментальних досліджень, можна стверджувати, що розроблений програмно-апаратний комплекс повністю відповідає вимогам до автоматизованих систем моніторингу біотехнологічних процесів. Комплексна перевірка підтвердила, що обрана архітектура, побудована на розподілі обчислювальних ресурсів між двома ядрами мікроконтролера ESP32-S3 під управлінням FreeRTOS, забезпечує високу надійність функціонування [39]. Розмежування задач збору даних та мережевої комунікації дозволило досягти детермінованої частоти дискретизації, що є критичним показником для наукових інструментів. Система продемонструвала здатність працювати в безперервному режимі тривалий час, автоматично відновлюючи працездатність після збоїв у мережевій інфраструктурі, що мінімізує ризики втрати даних під час довготривалих експериментів з культивування клітин.

Ключовим фактором, що визначив високу ефективність системи, стало впровадження гібридного протоколу передачі даних. Порівняльний аналіз показав, що перехід від текстового формату JSON до бінарної серіалізації телеметрії дозволив скоротити обсяг мережевого трафіку для пакетів потокових даних [20]. Це рішення не лише знизило навантаження на канал зв'язку, але й суттєво зменшило використання процесорного часу мікроконтролера на формування пакетів, що позитивно вплинуло на енергоефективність пристрою. Водночас збереження формату JSON для передачі рідкісних конфігураційних команд та повідомлень Discovery забезпечило необхідну гнучкість та простоту інтеграції нових типів сенсорів без зміни низькорівневого коду.

Оцінка швидкодії клієнтського ПЗ підтвердила правильність вибору стека технологій Python та PyQt5. Реалізація асинхронної моделі взаємодії, де мережевий клієнт винесено в окремий потік, а оновлення графічного інтерфейсу обмежено таймером (throttling), дозволила забезпечити плавну візуалізацію даних навіть при високій частоті їх надходження. Алгоритми буферизації та оптимізованого перемальовування графіків (set_data)

продемонстрували високу ефективність, споживаючи мінімальні ресурси персонального комп'ютера. Це робить розроблене програмне забезпечення придатним для використання навіть на малопотужних офісних ПК або ноутбуках, які зазвичай використовуються в лабораторних умовах.

Практична цінність отриманих результатів полягає у створенні відкритої та масштабованої платформи для досліджень ЛнЧ. На відміну від закритих пропрієтарних рішень, розроблена система дозволяє дослідникам самостійно модифікувати алгоритми обробки даних, додавати нові сенсори та інтегрувати систему з іншими лабораторними інструментами. Точність вимірювань, підтверджена серією тестів із еталонними впливами, є достатньою для вирішення задач моніторингу остеогенних процесів, а можливість дистанційного керування та логування даних значно спрощує рутинну роботу оператора [40].

5 РОЗРОБКА СТАРТАП-ПРОЕКТУ

5.1 Опис ідеї стартап-проекту

В межах даного підпункту проаналізовано зміст ідеї стартап-проекту, можливі напрямки застосування розробленого продукту та основні вигоди для потенційного користувача. Узагальнені дані наведено в таблиці 5.1.

Таблиця 5.1 — Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосувань	Вигоди для користувача
Створення доступного програмно-апаратного комплексу для автоматизації вимірювань у ЛнЧ. Система складається з апаратного модуля керування (на базі ESP32-S3) та кросплатформного програмного забезпечення (Python/PyQt5). Продукт забезпечує збір даних із сенсорів, керування умовами культивування та візуалізацію процесів (зокрема остеогенезу) у реальному часі через бездротовий зв'язок.	Науково-дослідні лабораторії: Проведення експериментів з тканинної інженерії, дослідження росту кісткової тканини та впливу імплантатів.	Дистанційний моніторинг: Можливість контролювати хід тривалих експериментів віддалено через Інтернет (MQTT).
	Фармакологічні дослідження: Скринінг лікарських препаратів на клітинних культурах (Drug screening) в автоматичному режимі.	Економія ресурсів: Автоматизація рутинних вимірювань вивільняє час персоналу та зменшує вплив людського фактору.
	Освітній сектор: Використання як навчального стенду для підготовки фахівців з біотехнологій та вбудованих систем.	Економічна ефективність: Значно менша вартість впровадження порівняно з промисловими інкубаторними станціями.
		Гнучкість: Відкрита архітектура дозволяє користувачу самостійно додавати нові сенсори та адаптувати ПЗ.

5.2 Технологічний аудит ідеї проекту

В межах даного підпункту проведено аудит технологічної здійсненності проекту та порівняння його характеристик із потенційними конкурентами. В якості аналогів для порівняння обрано:

Конкурент 1: Промислові станції моніторингу (наприклад, Zeiss, Nikon) — високоточні, але дорогі закриті системи.

Конкурент 2: Аматорські рішення (DIY) на базі Arduino/Serial — дешеві, але з обмеженим функціоналом і без повноцінного інтерфейсу.

Результати аудиту наведено в таблиці 5.2.

Таблиця 5.2 — Визначення сильних, слабких та нейтральних характеристик ідеї проекту

	Техніко-економічні характ. ідеї	(потенційні) товари/концепції конкурентів			W (Слабка)	N (Нейтр)	S (Сильна)
		Мій проект (ESP32 + Python)	Конкурент 1 (Промислові станції)	Конкурент 2 (Arduino DIY)			
1	Вартість реалізації	Низька (доступні компоненти)	Дуже висока (>$10 000)	Низька			+
2	Протокол передачі	MQTT (Binary) — швидкий, економний	Пропрієтарний / Закритий	HTTP / Serial — повільний			+
3	Графічний інтерфейс	PC (PyQt5) — зручний, графіки	PC (Special SW) — професійний	Відсутній або примітивний (LCD)		+	
4	Можливість модифікації	Відкритий код (Open Source)	Закрита архітектура	Відкритий код			+
5	Дистанційний доступ	Так (Wi-Fi / Internet)	Так (Хмарні сервіси)	Ні (Локальне підключення)			+
6	Точність вимірювань	Середня (залежить від сенсорів)	Висока (сертифікована)	Середня/ Низька	+		
7	Швидкість розгортання	Середня (потребує налаштування)	Висока (Plug-and-Play)	Низька (потребує пайки/коду)		+	

5.3 Аналіз ринкових можливостей запуску стартапу

Успішність запуску проекту залежить від правильної оцінки ринкової ситуації. Попередня характеристика потенційного ринку наведена в таблиці 5.3 [41].

Таблиця 5.3 — Попередня характеристика потенційного ринку стартап-проекту

№	Показники ринку (найменування)	Характеристика
1	Ринок	Ринок лабораторного обладнання для мікрофлюїдики та технологій «Organ-on-a-Chip».
2	Динаміка ринку	Зростаючий. Спостерігається стійкий тренд на автоматизацію in vitro досліджень для зменшення тестування на тваринах [42].
3	Рівень конкуренції	Середній. У сегменті промислового обладнання конкуренція висока (великі бренди), у сегменті бюджетних рішень для навчання та R&D — низька.
4	Бар'єри входу	Середні. Основний бар'єр — необхідність підтвердження точності вимірювань. Для наукових цілей бар'єри нижчі, ніж для медичної діагностики.
5	Специфіка вимог	Критичні вимоги до стабільності роботи (система має працювати без збоїв тижнями) та зручності експорту даних для наукових публікацій.

Надалі проведено аналіз потенційних клієнтів, їхніх потреб та вимог до продукту. Результати аналізу наведено в таблиці 5.4

Таблиця 5.4 — Характеристика потенційних клієнтів стартап-проекту

№	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Автоматизація досліджень при обмеженому фінансуванні	Наукові лабораторії університетів, аспіранти	Схильні до самостійної адаптації обладнання, шукають рішення з відкритим кодом (Open Source), чутливі до ціни.	Низька вартість, можливість програмування (Python API), доступ до сирих даних.
2	Прискорення розробки та виведення продуктів на ринок	Малі приватні R&D центри, біотех-стартапи	Готові платити більше за готове рішення "з коробки", щоб зекономити час персоналу. Цінують технічну підтримку.	Надійність роботи 24/7, зручний графічний інтерфейс, автоматичне формування звітів (CSV).
3	Наочна демонстрація біологічних процесів	Освітні заклади (Кафедри біоінженерії)	Закупівля обладнання партіями для навчальних класів. Орієнтуються на простоту використання студентами.	Інтуїтивно зрозумілий інтерфейс, безпека використання

5.4 Розроблення ринкової стратегії проекту

На основі проведеного аналізу ринкового середовища та оцінки сильних і слабких сторін проекту розроблено стратегію виведення продукту на ринок. Враховуючи специфіку цільової аудиторії, яка складається переважно з наукових та освітніх установ з обмеженим бюджетом, для реалізації стартап-проекту обрано стратегію диференціації. Суть даної стратегії полягає у

формуванні унікальної ціннісної пропозиції, яка відрізняє розроблений продукт від існуючих аналогів не лише за ціною, але й за набором функціональних можливостей та філософією використання.

Ключовим елементом стратегії є позиціонування системи як «відкритого інструменту» (Open Source / Open Hardware). На відміну від пропрієтарних промислових рішень, які є «чорними скриньками» для користувача, розроблений комплекс надає дослідникам повний доступ до прикладних програмних інтерфейсів та вихідного коду. Це дозволяє адаптувати алгоритми обробки даних під специфічні задачі конкретного експерименту, що є критично важливою перевагою для науково-дослідних робіт. Додатковим фактором диференціації виступає використання оптимізованих протоколів передачі даних (бінарний MQTT), що дозволяє розгортати систему на базі існуючої IT-інфраструктури лабораторії без необхідності закупівлі спеціалізованого серверного обладнання.

Стратегічне позиціонування продукту спрямоване на зайняття ніші доступного професійного обладнання для автоматизації рутинних біологічних досліджень. Система пропонується як економічно ефективна альтернатива ручному збору даних, що дозволяє підвищити точність експериментів та вивільнити час кваліфікованого персоналу. Конкурентна поведінка проекту базується на фланговій атаці лідерів ринку: замість прямої конкуренції у сегменті високоточного діагностичного обладнання, стартап фокусується на сегменті попередніх досліджень та навчання, де вимоги до метрологічної сертифікації є нижчими, а чутливість до ціни та зручності інтерфейсу — вищою. Такий підхід дозволяє мінімізувати бар'єри входу на ринок та сформувати лояльну базу користувачів серед молодих науковців та студентів.

5.5 Розроблення маркетингової програми

Для забезпечення ефективної комерціалізації розробленого продукту сформовано маркетингову програму, що базується на концепції «4P» та охоплює товарну, цінову, збутову та комунікаційну політики. Товарна політика стартапу передбачає виведення на ринок програмно-апаратного комплексу, що складається з двох взаємодоповнюючих компонентів. Першим компонентом є апаратний модуль, який включає мікроконтролер ESP32-S3 з попередньо встановленим вбудованим програмним забезпеченням та набір сумісних сенсорів. Другим компонентом є кросплатформне клієнтське програмне забезпечення для ПК, яке виступає основним інструментом взаємодії користувача з системою. Ключовою маркетинговою перевагою продукту визначено його модульність та відкритість, що дозволяє користувачеві адаптувати систему під конкретні наукові задачі, на відміну від жорстко стандартизованих конкурентних рішень.

Цінова політика проекту орієнтована на швидке проникнення на ринок шляхом встановлення доступної вартості, що є критичним фактором для бюджетних наукових установ та освітніх закладів. Для формування ціни апаратної частини обрано витратний метод, який передбачає встановлення фіксованої націнки на рівні 30-40% до собівартості компонентів та витрат на збірку. Програмне забезпечення розповсюджується за гібридною моделлю: базова версія з відкритим вихідним кодом надається безкоштовно для некомерційного використання, що сприяє популяризації платформи, тоді як розширені послуги з налаштування, інтеграції та технічної підтримки надаються на платній основі для комерційних лабораторій.

Політика розподілу (збуту) базується на використанні прямих каналів продажів, що дозволяє мінімізувати посередницьку націнку та забезпечити безпосередній контакт із споживачем. Основним майданчиком для реалізації продукції виступає спеціалізований веб-ресурс проекту, через який здійснюється прийом замовлень та надання доступу до документації. Додатковим каналом збуту є розміщення продукції на міжнародних маркетплейсах для електроніки та DIY-компонентів, орієнтованих на інженерну аудиторію. У перспективі передбачається налагодження

партнерських відносин з профільними кафедрами університетів для централізованого постачання навчальних комплектів у рамках освітніх програм з біотехнологій.

Комунікаційна політика спрямована на формування довіри до продукту в професійному середовищі. Основними інструментами просування обрано контент-маркетинг та участь у галузевих заходах. Стратегія просування включає публікацію статей у науково-технічних виданнях з демонстрацією реальних результатів досліджень, отриманих за допомогою розробленої системи, що слугує доказом її ефективності. Важливим елементом є розвиток спільноти користувачів на платформах GitHub та тематичних форумах, де розробники можуть демонструвати приклади коду та надавати консультації, формуючи імідж експертності та відкритості проекту.

5.6 Висновки до розділу

У п'ятому розділі магістерської дисертації проведено комплексне техніко-економічне обґрунтування розробки автоматизованої системи керування для ЛнЧ як інноваційного стартап-проекту. Результати технологічного аудиту продемонстрували, що запропоноване рішення на базі мікроконтролера ESP32 та відкритого ПЗ має суттєві конкурентні переваги перед існуючими промисловими аналогами [43]. Основними факторами успіху визначено значно нижчу собівартість реалізації, гнучкість архітектури та використання сучасних енергоефективних протоколів передачі даних.

Аналіз ринкового середовища підтвердив наявність стійкого попиту на бюджетні засоби автоматизації біотехнологічних досліджень, особливо в сегменті університетських лабораторій та невеликих R&D центрів, які стикаються з обмеженням фінансування. Розроблена ринкова стратегія, що базується на диференціації продукту через відкритість коду та модульність

конструкції, дозволяє мінімізувати бар'єри входу на ринок та уникнути прямої конкуренції з великими виробниками медичного обладнання.

Сформована маркетингова програма, орієнтована на прямі канали збуту та просування через професійні спільноти, забезпечує можливість органічного зростання проекту з мінімальними витратами на рекламу. Виявлені слабкі сторони проекту, зокрема відсутність метрологічної сертифікації, враховані у стратегії позиціонування продукту як інструменту для попередніх наукових досліджень та навчання, а не для клінічної діагностики. Таким чином, проведений аналіз свідчить про економічну доцільність реалізації розробленої системи та наявність високого потенціалу для її комерціалізації.

ВИСНОВКИ

У магістерській дисертації вирішено актуальне науково-прикладне завдання розробки автоматизованої системи керування та моніторингу для біотехнологічних пристроїв типу «Лабораторія на чипі». Проведений аналіз предметної області показав, що існуючі комерційні рішення для дослідження остеогенезу є високовартісними та складними для адаптації, що обґрунтувало доцільність створення власної відкритої платформи на базі мікроконтролера ESP32-S3 та мови програмування Python. На основі цього було спроєктовано архітектуру програмно-апаратного комплексу та розроблено структурно-функціональну схему підключення сенсорів мікроклімату, освітленості та виконавчих механізмів з використанням інтерфейсів I2C та GPIO.

Ключовим етапом роботи стала реалізація вбудованого програмного забезпечення мікроконтролера під управлінням операційної системи реального часу FreeRTOS. Ефективний розподіл навантаження між двома ядрами процесора дозволив ізолювати критичну до часу задачу вимірювання від роботи мережевого стека, що забезпечило стабільну частоту дискретизації. Для оптимізації інформаційного обміну було організовано гібридний протокол на базі MQTT: впровадження бінарного формату пакетів дозволило зменшити обсяг мережевого трафіку порівняно з текстовим JSON, суттєво підвищивши пропускну здатність каналу.

Практична реалізація клієнтського програмного забезпечення здійснена з використанням бібліотеки PyQt5. Застосування асинхронної архітектури з винесенням мережевого клієнта в окремий потік гарантувало високу чуйність графічного інтерфейсу, а розроблені алгоритми динамічної візуалізації та логування забезпечили зручність аналізу даних у реальному часі. Експериментальні дослідження на лабораторному макеті підтвердили працездатність системи, стабільність передачі даних та надійне відновлення з'єднання після збоїв. Техніко-економічний аналіз проекту довів його комерційну привабливість для університетських лабораторій та центрів

завдяки низькій вартості та відкритій архітектурі. Таким чином, мета роботи досягнута, а розроблена система готова до практичного використання в біомедичних дослідженнях.

ПЕРЕЛІК ПОСИЛАНЬ

1. Whitesides, George M. “The Origins and the Future of Microfluidics.” Nature, vol. 442, no. 7101, 2006, pp. 368–373, doi:10.1038/nature05058.
2. Sackmann, Eric K., et al. “The Present and Future Role of Microfluidics in Biomedical Research.” Nature, vol. 507, no. 7491, 2014, pp. 181–189, doi:10.1038/nature13118.
3. Mark, Daniel, et al. “Microfluidic Lab-on-a-Chip Platforms: Requirements, Characteristics and Applications.” Chemical Society Reviews, vol. 39, no. 3, 2010, pp. 1153–1182, doi:10.1039/b820557b.
4. Bhatia, Sangeeta N., and Donald E. Ingber. “Microfluidic Organs-on-Chips.” Nature Biotechnology, vol. 32, no. 8, 2014, pp. 760–772, doi:10.1038/nbt.2989.
5. Low, Lucie A., et al. “Organs-on-Chips: Into the next Decade.” Nature Reviews. Drug Discovery, vol. 20, no. 5, 2021, pp. 345–361, doi:10.1038/s41573-020-0079-3.
6. Au, Anthony K., et al. “3D-Printed Microfluidics.” Angewandte Chemie (International Ed. in English), vol. 55, no. 12, 2016, pp. 3862–3881, doi:10.1002/anie.201504382.
7. Ren, Kangning, et al. “Materials for Microfluidic Chip Fabrication.” Accounts of Chemical Research, vol. 46, no. 11, 2013, pp. 2396–2406, doi:10.1021/ar300314s.
8. McDonald, J. Cooper, and George M. Whitesides. “Poly(Dimethylsiloxane) as a Material for Fabricating Microfluidic Devices.” Accounts of Chemical Research, vol. 35, no. 7, 2002, pp. 491–499, doi:10.1021/ar010110q.
9. Halldorsson, Skarphedinn, et al. “Advantages and Challenges of Microfluidic Cell Culture in Polydimethylsiloxane Devices.” Biosensors & Bioelectronics, vol. 63, 2015, pp. 218–231, doi:10.1016/j.bios.2014.07.029.
10. Byun, Chang Kyu, et al. “Pumps for Microfluidic Cell Culture: Microfluidics and Miniaturization.” Electrophoresis, vol. 35, no. 2–3, 2014, pp. 245–257, doi:10.1002/elps.201300205.

11. Mansoorifar, Amin, et al. “Bone-on-a-Chip: Microfluidic Technologies and Microphysiologic Models of Bone Tissue.” Advanced Functional Materials, vol. 31, no. 6, 2021, p. 2006796, doi:10.1002/adfm.202006796.
12. Pires, Nuno Miguel Matos, et al. “Recent Developments in Optical Detection Technologies in Lab-on-a-Chip Devices for Biosensing Applications.” Sensors (Basel, Switzerland), vol. 14, no. 8, 2014, pp. 15458–15479, doi:10.3390/s140815458.
13. Grieshaber, Dorothee, et al. “Electrochemical Biosensors - Sensor Principles and Architectures.” Sensors (Basel, Switzerland), vol. 8, no. 3, 2008, pp. 1400–1458, doi:10.3390/s80314000.
14. Benson, Kathrin, et al. “Impedance-Based Cell Monitoring: Barrier Properties and Beyond.” Fluids and Barriers of the CNS, vol. 10, no. 1, 2013, p. 5, doi:10.1186/2045-8118-10-5.
15. Mou, Lei, et al. “Integrated Biosensors for Monitoring Microphysiological Systems.” Lab on a Chip, vol. 22, no. 20, 2022, pp. 3801–3816, doi:10.1039/d2lc00262k.
16. Adeagbo, Adesunmbo Adeboye. “IOT Based Environment Monitoring System Using.” arXiv [Eess.SY], 2024, doi:10.48550/ARXIV.2405.14047
17. Espressif Systems. ESP32-S3 Series Datasheet. Ver. 1.1, Espressif Systems, 2022, documentation.espressif.com/esp32-s3_datasheet_en.pdf. Accessed 30 Oct. 2025.
18. Espressif Systems. ESP32-S3 Technical Reference Manual. Ver. 1.1, Espressif Systems, 2023, documentation.espressif.com/esp32-s3_technical_reference_manual_en.pdf. Accessed 30 Oct. 2025.
19. Oliphant, Travis E. “Python for Scientific Computing.” Computing in Science & Engineering, vol. 9, no. 3, 2007, pp. 10–20, doi:10.1109/mcse.2007.58.
20. Naik, Nitin. “Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP.” 2017 IEEE International Systems Engineering Symposium (ISSE), IEEE, 2017.

21. Banks, Andrew, and Rahul Gupta, editors. MQTT Version 3.1.1. OASIS, 29 Oct. 2014, docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. Accessed 30 Oct. 2025.
22. "Struct — Interpret Bytes as Packed Binary Data." Python 3.10 Documentation, Python Software Foundation, docs.python.org/3/library/struct.html. Accessed 30 Oct. 2025.
23. Hunter, John D. “Matplotlib: A 2D Graphics Environment.” Computing in Science & Engineering, vol. 9, no. 3, 2007, pp. 90–95, doi:10.1109/mcse.2007.55.
24. "Pyreverse." Pylint Documentation, PyCQA, pylint.pycqa.org/en/latest/pyreverse.html. Accessed 30 Oct. 2025.
25. McCabe, T. J. “A Complexity Measure.” IEEE Transactions on Software Engineering, vol. SE-2, no. 4, 1976, pp. 308–320, doi:10.1109/tse.1976.233837.
26. Crockford, Douglas. “RFC 4627: The Application/Json Media Type for JavaScript Object Notation (JSON).” IETF Datatracker, datatracker.ietf.org/doc/html/rfc4627.
27. Barry, Richard. Mastering the FreeRTOS Real Time Kernel: A Hands-On Tutorial Guide. Real Time Engineers Ltd., 2016, www.freertos.org/media/2018/161204_Mastering_the_FreeRTOS_Real_Time_Kernel-A_Hands-On_Tutorial_Guide.pdf. Accessed 30 Oct. 2025.
28. Ahmad, Aisyah Binti, and Setiawan Prabowo. Real Time Operating Systems for Embedded Applications: Design and Implementation. doi:10.31838/JIVCT/02.01.05.
29. Tanenbaum, Andrew S., and Herbert Bos. Modern Operating Systems. 4th ed., Pearson, 2015, csc-knu.github.io/sys-prog/books/Andrew%20S.%20Tanenbaum%20-%20Modern%20Operating%20Systems.pdf. Accessed 30 Oct. 2025.
30. "High-Performance Real-Time Data Visualization with Python and Qt." YouTube, uploaded by Enthought, 16 July 2014, www.youtube.com/watch?v=s5R-1g5l5H8. Accessed 30 Oct. 2025.

31. Datar, Mayur, et al. "Maintaining Stream Statistics over Sliding Windows." SIAM Journal on Computing, vol. 31, no. 6, 2002, pp. 1794-1813, doi.org/10.1137/S0097539701398363.
32. "Pandas Documentation." Pandas, NumFOCUS, pandas.pydata.org/pandas-docs/stable/. Accessed 30 Oct. 2025.
33. "HiveMQ: The Enterprise MQTT Platform." HiveMQ, www.hivemq.com/. Accessed 30 Oct. 2025.
34. "MQTT Explorer." MQTT Explorer, mqtt-explorer.com/. Accessed 30 Oct. 2025.
35. Artusi, Alessandro, et al. Image Content Retargeting: Maintaining Color, Tone, and Spatial Consistency. Taylor & Francis, 2021, doi:10.1201/9781315372624.
36. BME280: Combined Humidity and Pressure Sensor. Bosch Sensortec, 2016. AllDatasheet, www.alldatasheet.com/datasheet-pdf/pdf/1132060/BOSCH/BME280.html. Accessed 30 Oct. 2025.
37. BH1750FVI: Digital 16bit Serial Output Type Ambient Light Sensor IC. ROHM Semiconductor, 2011. AllDatasheet, www.alldatasheet.com/datasheet-pdf/pdf/338083/ROHM/BH1750FVI.html. Accessed 30 Oct. 2025.
38. WS2812B Intelligent Control LED Integrated Light Source. Worldsemi, 2015. AllDatasheet, www.alldatasheet.com/datasheet-pdf/pdf/1179113/WORLDSEMI/WS2812B.html. Accessed 30 Oct. 2025.
39. Lee, In, and Kyoochun Lee. “The Internet of Things (IoT): Applications, Investments, and Challenges for Enterprises.” Business Horizons, vol. 58, no. 4, 2015, pp. 431–440, doi:10.1016/j.bushor.2015.03.008.
40. Young, Edmond W. K., and David J. Beebe. “Fundamentals of Microfluidic Cell Culture in Controlled Microenvironments.” Chemical Society Reviews, vol. 39, no. 3, 2010, pp. 1036–1048, doi:10.1039/b909900j.
41. Organs-on-Chips Market and Technology Landscape 2019. Yole Développement, 2019, medias.yolegroup.com/uploads/2019/10/Yole_YD19046_-Organs-on-chips-Market-and-Technology-Landscape-2019_sample.pdf. Accessed 30 Oct. 2025.

42. Franzen, Nora, et al. “Impact of Organ-on-a-Chip Technology on Pharmaceutical R&D Costs.” Drug Discovery Today, vol. 24, no. 9, 2019, pp. 1720–1724, doi:10.1016/j.drudis.2019.06.003.
43. Zhang, Yu Shrike, et al. “Multisensor-Integrated Organs-on-Chips Platform for Automated and Continual in Situ Monitoring of Organoid Behaviors.” Proceedings of the National Academy of Sciences of the United States of America, vol. 114, no. 12, 2017, pp. E2293–E2302, doi:10.1073/pnas.1612906114.