ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Програмно-апаратна система аналізу якості повітря на основі мікроконтролерів»

на здобуття освітнього ступеня бакалавра
зі спеціальності 123 – Комп'ютерна інженерія
(код, найменування спеціальності)
освітньо-професійної програми Комп'ютерна інженерія
(назва)

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

_____ *Микола ЗАЛЕВСЬКИЙ*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. КІД-41
Микола ЗАЛЕВСЬКИЙ
Ім'я, ПРІЗВИЩЕ
Керівник: Сергій КОРОТКОВ
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ
Рецензент: _____
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Комп'ютерної інженерії

Ступінь вищої освіти бакалавр

Спеціальність 123 – Комп'ютерна інженерія

Освітньо-професійна програма Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедрою Наталія ЛАЩЕВСЬКА
_____Ім'я, ПРІЗВИЩЕ
«_____» _____ 2025р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Залевський Микола Олегович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Програмно-апаратна система аналізу якості повітря на основі мікроконтролерів
керівник кваліфікаційної роботи Коротков Сергій Станіславович,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025р. № 56
2. Строк подання кваліфікаційної роботи «5» червня 2025р.
3. Вихідні дані до кваліфікаційної роботи:
   - Мікроконтролери сімейства ESP (зокрема ESP32).
   - Різні типи сенсорів якості повітря (наприклад, для вимірювання PM2.5, PM10, CO, CO2, VOC).
   - Протоколи бездротового зв'язку: Wi-Fi (IEEE 802.11 b/g/n).
   - Протоколи передачі даних: HTTP, MQTT (розглянути можливість використання).
   - Формати даних: JSON (для передачі даних).
   - Хмарні платформи для збору та візуалізації даних (розглянути можливість використання безкоштовних сервісів).
   - Інтерфейси підключення сенсорів: GPIO, I2C, UART.
   - Живлення системи: USB, автономне живлення (батареї).

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

– Аналіз існуючих систем та технологій моніторингу якості повітря.

– Обґрунтування вибору апаратної платформи (мікроконтролера ESP32) та сенсорів якості повітря.

– Розробка архітектури програмно-апаратної системи аналізу якості повітря.

– Розробка програмного забезпечення для мікроконтролера ESP32 (збір даних з сенсорів та їх передача).

– Розробка (опис) програмного забезпечення для серверної частини (збір, обробка та зберігання даних).

– Розробка (опис) інтерфейсу візуалізації даних про якість повітря.

– Оцінка ефективності та потенційні результати роботи розробленої системи.

5. Перелік ілюстративного матеріалу: *презентація*

6. Дата видачі завдання «24» лютого 2025р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
	Збір даних	7	Виконано
	Обробка матеріалу	1	Виконано
	Розробка теоретичної частини	8	Виконано
	Розробка моделі мережі	5	Виконано
	Висновки за результатами аналізу	3	Виконано
	Оформлення пояснювальної записки	2	Виконано
	Розробка презентації	4	Виконано
	Передзахист		Виконано
	Здача в деканат		Виконано

Здобувач вищої освіти _____ Микола ЗАЛЕВСЬКИЙ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Сергій КОРОТКОВ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 40 стор., 17 рис., 1 табл., 20 джерел.

Мета роботи – Розробити концепцію програмно-апаратної системи для аналізу якості повітря на основі мікроконтролерів та визначити основні технічні рішення для її реалізації.

Об'єкт дослідження – Процес моніторингу та аналізу якості повітря за допомогою електронних систем.

Предмет дослідження – Архітектура програмно-апаратної системи, сенсори якості повітря, мікроконтролери та методи передачі та обробки даних для аналізу якості повітря.

Короткий зміст роботи: У кваліфікаційній роботі розглянуто сучасний стан проблеми забруднення повітря та обґрунтовано актуальність розробки ефективних систем моніторингу. Проведено аналіз існуючих систем моніторингу якості повітря, їх переваги та недоліки. Особливу увагу приділено аналізу різних типів сенсорів якості повітря, їхнім характеристикам та принципам роботи, а також огляду мікроконтролерів, придатних для використання в таких системах, зокрема ESP32.

Розроблено концептуальну архітектуру програмно-апаратної системи, що включає мережу датчиків на базі мікроконтролерів ESP32, передачу даних на сервер та програмне забезпечення для збору, обробки та візуалізації отриманої інформації. Описано можливі протоколи передачі даних та підходи до організації бази даних. Розглянуто метрики для оцінки ефективності роботи розробленої системи та можливі сценарії її тестування.

Результати роботи демонструють потенційну ефективність запропонованої програмно-апаратної системи для здійснення розподіленого та оперативного моніторингу якості повітря.

КЛЮЧОВІ СЛОВА: ЯКІСТЬ ПОВІТРЯ, МОНІТОРИНГ, СЕНСОРИ ЯКОСТІ ПОВІТРЯ, МІКРОКОНТРОЛЛЕРИ, ESP32, ПРОГРАМНО-АПАРАТНА СИСТЕМА, ПЕРЕДАЧА ДАНИХ, ВІЗУАЛІЗАЦІЯ ДАНИХ, МЕТРИКИ ОЦІНКИ.

ЗМІСТ

ВСТУП 6
1 АНАЛІЗ ІСНУЮЧИХ СИСТЕМ ТА ТЕХНОЛОГІЙ МОНІТОРИНГУ ЯКОСТІ ПОВІТРЯ 8
1.1 Сучасний стан проблеми забруднення повітря та актуальність його моніторингу 8
1.2 Огляд існуючих систем моніторингу якості повітря 10
1.3 Аналіз використовуваних сенсорів якості повітря 13
1.4 Огляд мікроконтролерів для застосування в системах моніторингу якості повітря 16
2 РОЗРОБКА ПРОГРАМНО-АПАРАТНОЇ СИСТЕМИ АНАЛІЗУ ЯКОСТІ ПОВІТРЯ НА ОСНОВІ МІКРОКОНТРОЛЕРІВ 22
2.1 Теоретичні основи побудови програмно апаратних систем моніторингу 22
2.2 Обгрунтування вибору апаратної платформи та сенсорів 25
2.3 Вибір конкретних моделей сенсорів 28
3 РОЗРОБКА ПРОГРАМНО-АПАРАТНОЇ СИСТЕМИ АНАЛІЗУ ЯКОСТІ ПОВІТРЯ НА ОСНОВІ МІКРОКОНТРОЛЕРІВ 34
3.1 Розробка схеми підключення сенсорів до мікроконтролера ESP32 34
3.2 Опис передачі даних на сервер 39
3.3 Програмне забезпечення для збору, обробки та візуалізації даних 41
ВИСНОВКИ 46
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 48
Додаток А 48
Додаток Б 58

ВСТУП

У сучасному світі проблема забруднення атмосферного повітря набуває дедалі більшої гостроти та стає однією з ключових екологічних загроз для здоров'я людини та сталого розвитку міст і регіонів. Зростання промисловості, інтенсифікація транспортних потоків та інші антропогенні фактори призводять до накопичення в повітрі шкідливих речовин, що негативно впливають на екосистеми та спричиняють різноманітні захворювання.

У цьому контексті ефективний та оперативний моніторинг якості повітря є критично важливим для своєчасного виявлення джерел забруднення, оцінки рівня небезпеки та прийняття обґрунтованих рішень щодо покращення екологічної ситуації.

Традиційні методи моніторингу якості повітря часто передбачають використання стаціонарних станцій, які є дорогими у встановленні та обслуговуванні, а також забезпечують лише дискретні дані в обмежених точках. Розвиток мікроконтролерних технологій та сенсорів якості повітря відкриває нові можливості для створення більш гнучких, масштабованих та економічно вигідних систем моніторингу.

Завдяки компактності, низькому енергоспоживанню та можливості бездротової передачі даних, мікроконтролери, такі як ESP32, стають перспективною платформою для розробки розподілених мереж датчиків, здатних здійснювати безперервний моніторинг якості повітря на значних територіях.

Актуальність даної кваліфікаційної роботи зумовлена необхідністю розробки ефективних та доступних інструментів для аналізу якості повітря. Створення програмно-апаратної системи на основі мікроконтролерів дозволить отримувати детальну інформацію про рівень забруднення в різних локаціях, оперативно реагувати на зміни та формувати більш повну картину екологічної ситуації.

Метою даної кваліфікаційної роботи є розробка концепції програмно-апаратної системи для аналізу якості повітря на основі мікроконтролерів.

Для досягнення поставленої мети передбачається вирішення наступних завдань:

- Провести аналіз існуючих систем та технологій моніторингу якості повітря;
- Вивчити характеристики та принципи роботи різних типів сенсорів якості повітря, придатних для використання з мікроконтролерами;
- Обґрунтувати вибір апаратної платформи (на прикладі ESP32) та сенсорів для розробки системи;
- Розробити архітектуру програмно-апаратної системи, включаючи схему підключення датчиків, алгоритм збору та передачі даних;
- Описати можливі підходи до програмного забезпечення для обробки та візуалізації отриманих даних;
- Визначити метрики для оцінки ефективності роботи розробленої системи та запропонувати можливі сценарії її тестування.

Очікується, що результати даної роботи стануть основою для подальшої розробки практично застосовних систем моніторингу якості повітря, сприяючи підвищенню екологічної обізнаності та прийняттю більше ефективних заходів для покращення стану довкілля.

1 АНАЛІЗ ІСНУЮЧИХ СИСТЕМ ТА ТЕХНОЛОГІЙ МОНІТОРИНГУ ЯКОСТІ ПОВІТРЯ

1.1 Сучасний стан проблеми забруднення повітря та актуальність його моніторингу

У XXI столітті проблема забруднення атмосферного повітря є однією з найсерйозніших екологічних та соціально-економічних проблем, що постали перед людством. Інтенсивний розвиток промисловості, енергетики, транспорту та сільського господарства призводить до викидів значної кількості шкідливих речовин в атмосферу, які негативно впливають на здоров'я людей, стан екосистем та клімат планети в цілому.

За даними Всесвітньої організації охорони здоров'я (ВООЗ), забруднення повітря є однією з основних причин передчасної смертності у світі, щорічно спричиняючи мільйони смертей від серцево-судинних та респіраторних захворювань, раку легень та інших недуг. Особливо вразливими до негативного впливу забрудненого повітря є діти, літні люди та особи з хронічними захворюваннями.

Основними джерелами забруднення повітря є промислові підприємства, транспорт, енергетика, сільське господарство та побутове опалення, кожен з яких робить свій внесок у викиди різноманітних забруднюючих речовин. Наслідки забруднення повітря є багатогранними, включаючи шкоду для здоров'я, погіршення стану екосистем, зниження врожайності, руйнування інфраструктури та зміни клімату.

Усвідомлюючи серйозність цієї проблеми, на національному та міжнародному рівнях розробляються та впроваджуються різноманітні програми та нормативи у сфері якості повітря.

На міжнародному рівні ключову роль відіграють такі організації та ініціативи:

- *Всесвітня організація охорони здоров'я (ВООЗ).* Розробляє глобальні рекомендації та стандарти щодо якості повітря, проводить моніторинг та оцінку впливу забруднення на здоров'я населення;
- *Програма Організації Об'єднаних Націй з навколишнього середовища (ЮНЕП).* Координує міжнародне співробітництво у сфері охорони атмосферного повітря, підтримує розробку національних стратегій та програм;
- *Конвенція про транскордонне забруднення повітря на великі відстані (CLRTAP).* У рамках Європейської економічної комісії ООН спрямована на боротьбу з транскордонним забрудненням повітря;
- *Цілі сталого розвитку ООН (ЦСР).* Зокрема, Ціль 3 ("Міцне здоров'я та добробут") та Ціль 11 ("Сталі міста та громади") включають завдання, пов'язані з покращенням якості повітря.

На національному рівні в Україні діють відповідні законодавчі та нормативно-правові акти, спрямовані на регулювання викидів забруднюючих речовин та контроль за якістю атмосферного повітря. До них належать:

- *Закон України "Про охорону атмосферного повітря"*. Визначає правові та організаційні основи діяльності у сфері охорони атмосферного повітря;
- *гігієнічні нормативи якості атмосферного повітря населених місць*. Встановлюють гранично допустимі концентрації (ГДК) основних забруднюючих речовин;
- *національні та регіональні програми щодо поліпшення стану атмосферного повітря*;
- *система державного моніторингу довкілля*, яка включає спостереження за якістю атмосферного повітря.

Однак, незважаючи на наявність цих програм та нормативів, проблема забруднення повітря в Україні залишається актуальною, особливо у великих промислових центрах. Ефективний моніторинг є ключовим елементом для забезпечення дотримання встановлених стандартів, оцінки ефективності вжитих

заходів та своєчасного реагування на випадки перевищення гранично допустимих концентрацій.

Таким чином, розробка та впровадження сучасних, ефективних та доступних систем моніторингу якості повітря, зокрема на основі мікроконтролерних технологій, є важливим інструментом для підтримки реалізації національних та міжнародних екологічних цілей та забезпечення здоров'я громадян.

1.2 Огляд існуючих систем моніторингу якості повітря

Існуючі системи моніторингу якості повітря можна класифікувати за різними критеріями, зокрема за їхнім призначенням, рівнем точності, охопленням території та вартістю. Основними типами таких систем є промислові, наукові та аматорські.

Промислові системи моніторингу якості повітря використовуються підприємствами для контролю викидів забруднюючих речовин у процесі виробництва та забезпечення дотримання екологічних норм і стандартів. Ці системи, як правило, є стаціонарними та встановлюються безпосередньо на джерелах викидів (наприклад, димові труби, вентиляційні системи) або на межі санітарно-захисної зони підприємства.

Основні характеристики промислових систем моніторингу:

- *Висока точність вимірювань.* Використовується високоточне аналітичне обладнання, здатне вимірювати концентрацію широкого спектра забруднювачів з високою роздільною здатністю;
- *Безперервний моніторинг.* Забезпечують постійний збір та аналіз даних у режимі реального часу;
- *Автоматизація процесів.* Включають автоматизовані системи пробопідготовки, калібрування та передачі даних до централізованих систем управління;

– *Відповідність нормативним вимогам.* Розробляються з урахуванням вимог національних та міжнародних екологічних стандартів;

– *Висока вартість.* Встановлення та обслуговування таких систем потребує значних фінансових витрат.

Приклади промислових систем моніторингу включають стаціонарні газоаналізатори, системи моніторингу твердих частинок, системи контролю викидів летких органічних сполук тощо.

Наукові системи моніторингу якості повітря використовуються дослідницькими установами, університетами та екологічними організаціями для проведення фундаментальних та прикладних досліджень у галузі атмосферного забруднення. Ці системи можуть бути як стаціонарними, так і мобільними, що дозволяє проводити вимірювання в різних локаціях та досліджувати просторово-часові закономірності поширення забруднювачів.

Основні характеристики наукових систем моніторингу:

– *висока гнучкість та адаптивність*. Системи можуть бути налаштовані для вимірювання широкого спектра параметрів, включаючи нові або маловивчені забруднюючі речовини;

– *використання передових технологій*. Часто включають новітні сенсорні технології, спектрометричне обладнання, метеорологічні станції та системи дистанційного зондування (наприклад, супутникові дані);

– *можливість проведення комплексних досліджень*. Забезпечують збір даних, необхідних для моделювання процесів забруднення, оцінки впливу на екосистеми та здоров'я людини;

– *вартість може варіюватися.* Залежить від складності обладнання та масштабів досліджень.

Приклади наукових систем моніторингу включають мережі автоматизованих станцій моніторингу атмосферного повітря, мобільні лабораторії для вимірювання забруднення в реальному часі, аерозольні монітори, системи аналізу атмосферних опадів тощо.

В останні роки набувають популярності *аматорські системи моніторингу якості повітря*, розроблені ентузіастами, громадськими організаціями або в рамках наукових проєктів із залученням громадян (citizen science). Ці системи, як правило, є більш доступними за вартістю та простішими у використанні, часто базуючись на недорогих сенсорах та мікроконтролерних платформах (наприклад, Arduino, ESP32).

Основні характеристики аматорських систем моніторингу:

- *Низька вартість.* Використання відносно недорогих компонентів;
- *Простота встановлення та використання.* Легкість розгортання та обслуговування;
- *Можливість створення розгалужених мереж.* Дозволяють охоплювати великі території за рахунок великої кількості датчиків;
- *Точність вимірювань* може бути нижчою порівняно з промисловими та науковими системами, потребує калібрування та порівняння з еталонними даними.
- *Потенціал для підвищення екологічної обізнаності.* Залучення громадян до процесу моніторингу сприяє підвищенню їхньої екологічної свідомості.

Приклади аматорських систем моніторингу включають саморобні датчики PM2.5 та PM10 на базі Arduino або ESP32, підключені до онлайн-платформ для візуалізації даних, проєкти з моніторингу якості повітря у шкільних громадах тощо.

Існуючі системи моніторингу якості повітря відрізняються за своїми характеристиками та призначенням. Промислові системи забезпечують високу точність для контролю викидів, наукові системи використовуються для глибоких досліджень, а аматорські системи пропонують доступну альтернативу для широкого охоплення території та залучення громадськості. Розробка програмно-апаратної системи на основі мікроконтролерів може поєднати переваги доступності та можливості створення розгалужених мереж, водночас прагнучи до забезпечення прийнятної точності вимірювань шляхом належного калібрування та обробки даних.

1.3 Аналіз використовуваних сенсорів якості повітря

Ефективний моніторинг якості повітря значною мірою залежить від характеристик використовуваних сенсорів. На ринку представлений широкий спектр сенсорів, що різняться за принципом дії, типом вимірюваних забруднювачів, точністю, чутливістю, діапазоном вимірювань, часом відгуку, енергоспоживанням та вартістю. Для розробки програмно-апаратної системи аналізу якості повітря на основі мікроконтролерів важливо обрати сенсори, які б поєднували прийнятну точність з низькою вартістю та енергоспоживанням.

Розглянемо основні типи сенсорів якості повітря, що можуть бути використані в таких системах:

Сенсори твердих частинок (Particulate Matter Sensors). Тверді частинки (PM10, PM2.5, PM1.0) є одними з найнебезпечніших забруднювачів повітря. Для їх вимірювання найчастіше використовуються оптичні сенсори (лазерні або світлодіодні).

а) *Принцип роботи.* Оптичні сенсори використовують джерело світла (лазер або світлодіод) для освітлення потоку повітря, що проходить через сенсор. Частинки пилу, що знаходяться в повітрі, розсіюють світло. Фотодетектор вимірює інтенсивність розсіяного світла, яка пропорційна концентрації твердих частинок певного розміру.

б) Характеристики:

1) *Типи вимірюваних частинок.* PM1.0, PM2.5, PM10 (залежно від моделі);

2) *Діапазон вимірювань.* Від 0 до сотень або тисяч мкг/м³;

3) *Точність.* Зазвичай ±10-15% або ± певне значення в мкг/м³; Точність може залежати від умов експлуатації (вологість, температура);

4) *Час відгуку.* Зазвичай кілька секунд;

5) *Енергоспоживання.* Від десятків до сотень мА;

6) *Вартість.* Відносно недорогі, особливо аматорські моделі;

в) *Приклади.* Сенсори серій PMS (Plantower), SDS (Nova Fitness).

Газові сенсори (Gas Sensors). Для вимірювання концентрації різних газів-забруднювачів використовуються різні типи сенсорів, серед яких найбільш поширені електрохімічні, металооксидні (MOS) та інфрачервоні (NDIR) сенсори.

а) Електрохімічні сенсори:

1) *Принцип роботи.* Ґрунтуються на електрохімічній реакції між газом-аналітом та електролітом усередині сенсора, що призводить до виникнення електричного струму, пропорційного концентрації газу;

2) *Характеристики.* Висока чутливість та селективність до певних газів (наприклад, CO, O2, NO2, SO2). Низьке енергоспоживання. Обмежений термін служби. Чутливість до температури та вологості;

3) *Приклади.* Сенсори серій MQ (для різних газів, хоча мають нижчу селективність), сенсори від Alphasense, Sensirion (деякі моделі);

б) Металооксидні (MOS) сенсори:

1) *Принцип роботи.* Зміна електричного опору напівпровідникового матеріалу (зазвичай оксиду металу, такого як SnO2, WO3) при адсорбції молекул газу на його поверхні;

2) *Характеристики.* Низька вартість, відносно просте використання, широкий діапазон чутливості до різних газів (CO, CH4, LPG, NH3, NOx). Можлива перехресна чутливість до різних газів. Потребують прогріву для стабільної роботи, що збільшує енергоспоживання. Точність може бути нижчою порівняно з електрохімічними сенсорами;

3) *Приклади.* Сенсори серій MQ (наприклад, MQ-135 для якості повітря), MiCS-5525.

в) Інфрачервоні (NDIR) сенсори:

1) *Принцип роботи.* Вимірювання поглинання інфрачервоного випромінювання певними газами (наприклад, CO2, CH4);

2) *Характеристики.* Висока точність та стабільність для вимірювання газів, що поглинають ІЧ-випромінювання. Менша чутливість

до інших газів. Вище енергоспоживання через необхідність використання ІЧ-джерела та детектора. Вища вартість;

3) *Приклади.* Сенсори від SenseAir, MH-Z19B (для CO2).

Сенсори летких органічних сполук (VOC Sensors). Леткі органічні сполуки (ЛОС) включають широкий спектр хімічних речовин, що можуть бути шкідливими для здоров'я. Для їх вимірювання часто використовуються MOS-сенсори або фотоіонізаційні детектори (PID) (останні дорожчі та рідше використовуються в аматорських системах):

- *MOS-сенсори (для ЛОС).* Подібні за принципом роботи до MOS-сенсорів для інших газів, але налаштовані на виявлення широкого спектра органічних сполук. Часто надають умовний індекс якості повітря (IAQ) або рівень загальних ЛОС (TVOC);
- *Характеристики.* Низька вартість, виявлення широкого спектра ЛОС, можлива перехресна чутливість, потребують прогріву. Точність може бути відносно невисокою;
- *Приклади.* Сенсори серій MQ, BME680 (комбінований сенсор з VOC), SGP30 (Sensirion).

Вибір сенсорів для програмно-апаратної системи. При виборі сенсорів для вашої системи необхідно враховувати:

- Типи забруднювачів, які ви плануєте моніторити;
- Необхідну точність вимірювань залежно від мети моніторингу;
- Бюджет проєкту;
- Енергоспоживання сенсорів, особливо якщо планується автономне живлення;
- Інтерфейс підключення сенсорів до мікроконтролера (аналоговий, цифровий I2C, SPI);
- Наявність бібліотек та прикладів коду для роботи з обраними сенсорами на платформі ESP32;

Для початкового етапу розробки часто обирають комбінацію оптичного сенсора твердих частинок (наприклад, PMS5003) та MOS-сенсора для загальної

оцінки якості повітря (наприклад, MQ-135 або SGP30), враховуючи їхню відносно низьку вартість та доступність.

1.4 Огляд мікроконтролерів для застосування в системах моніторингу якості повітря

Мікроконтролери є ключовим елементом програмно-апаратних систем моніторингу якості повітря, оскільки вони відповідають за збір даних з сенсорів, їхню попередню обробку, керування роботою периферійних пристроїв (наприклад, модулів зв'язку) та передачу інформації на центральний сервер або інші пристрої. При виборі мікроконтролера для такої системи необхідно враховувати ряд важливих характеристик, серед яких:

- *Продуктивність.* Тактова частота процесора та обсяг оперативної пам'яті (RAM) визначають здатність мікроконтролера обробляти дані в реальному часі та виконувати складні алгоритми;
- *Кількість та типи інтерфейсів.* Наявність достатньої кількості аналогових входів (ADC) для підключення аналогових сенсорів, цифрових входів/виходів (GPIO) для керування іншими компонентами, а також комунікаційних інтерфейсів (UART, SPI, I2C) для зв'язку з сенсорами та іншими модулями;
- *Можливості бездротового зв'язку.* Для систем, що передбачають передачу даних бездротовим шляхом, важлива наявність вбудованих модулів Wi-Fi та/або Bluetooth;
- *Енергоспоживання.* Для систем, що працюють від батарей або передбачають тривалий час автономної роботи, низьке енергоспоживання є критично важливим;
- *Вартість.* Вартість мікроконтролера впливає на загальну вартість системи, особливо при масовому виробництві;

– *Екосистема та підтримка.* Наявність розвиненої екосистеми (інструменти розробки, бібліотеки, приклади коду, активна спільнота) значно полегшує процес розробки та налагодження;

– *Розмір та форм-фактор.* Компактність може бути важлива для розміщення датчиків у важкодоступних місцях або для створення портативних пристроїв.

Розглянемо деякі з найбільш популярних мікроконтролерів, які часто використовуються в системах моніторингу якості повітря:

а) ESP32:

1) *Виробник.* Espressif Systems;

2) *Основні характеристики.* Двоядерний 32-бітний процесор Tensilica LX6 (до 240 МГц), вбудована пам'ять ROM та SRAM, вбудовані модулі Wi-Fi (802.11 b/g/n) та Bluetooth (Classic та BLE), велика кількість GPIO, ADC, DAC, UART, SPI, I2C, I2S, низьке енергоспоживання в режимі сну, компактні розміри та низька вартість, активна спільнота та широка підтримка інструментів розробки (Arduino IDE, ESP-IDF). Типову плату розробки на базі мікроконтролера ESP32 (ESP32-WROOM-32 Dev Kit) (рис. 1.1);

3) *Переваги для систем моніторингу якості повітря.* Вбудований Wi-Fi забезпечує легку передачу даних на сервер. Достатня кількість інтерфейсів для підключення різних сенсорів. Низьке енергоспоживання дозволяє використовувати батарейне живлення;

4) *Недоліки.* Споживання струму в активному режимі може бути вищим порівняно з деякими іншими мікроконтролерами.

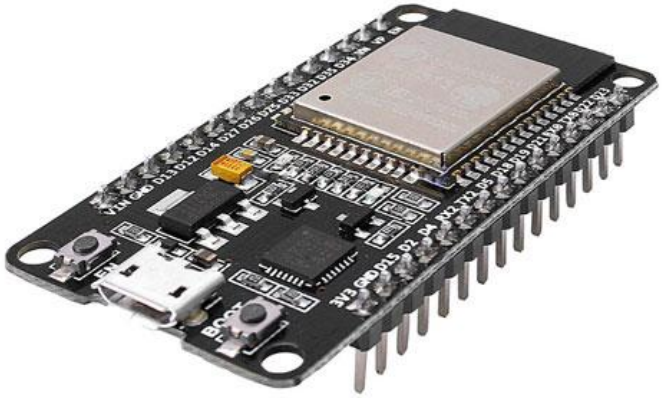


Рисунок 1.1 – ESP32-WROOM-32 Dev Kit

б) ESP8266:

1) *Виробник.* Espressif Systems;

2) *Основні характеристики.* 32-бітний процесор Tensilica L106 (до 80 МГц), вбудована пам'ять ROM та SRAM, вбудований модуль Wi-Fi (802.11 b/g/n), менша кількість GPIO та ADC порівняно з ESP32, низька вартість та компактні розміри, широка підтримка в Arduino IDE. Популярну плату розробки NodeMCU на базі мікроконтролера ESP8266 (рис. 1.2);

3) *Переваги для систем моніторингу якості повітря.* Вбудований Wi-Fi за низької вартості. Достатньо для простих систем з невеликою кількістю сенсорів;

4) *Недоліки.* Менша продуктивність та менша кількість периферійних інтерфейсів порівняно з ESP32. Відсутність вбудованого Bluetooth.

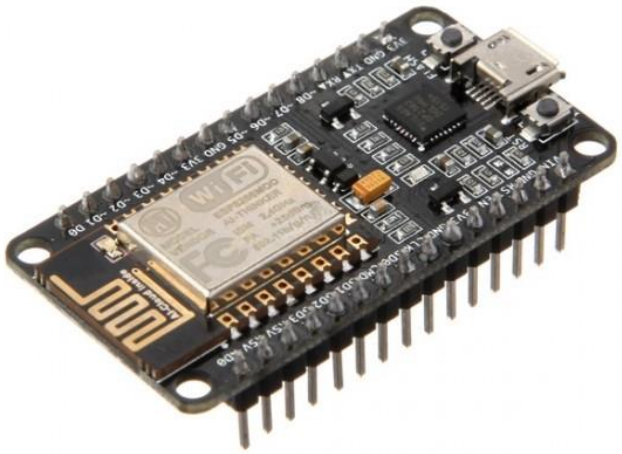


Рисунок 1.2 – Плата розробки NodeMCU на базі мікроконтролера ESP8266

в) *Arduino* (на базі ATmega328P та інших мікроконтролерів):

1) *Виробник.* Arduino LLC та інші;

2) *Основні характеристики.* Різні моделі з різними характеристиками (наприклад, Arduino Uno на базі ATmega328P має 8-бітний процесор, обмежену кількість RAM та Flash, аналогові та цифрові піни). Для підключення до мережі потрібні зовнішні модулі (наприклад, ESP8266 або Ethernet shield). Типову плату Arduino Uno на базі мікроконтролера ATmega328P (рис. 1.3);

3) *Переваги для систем моніторингу якості повітря.* Величезна спільнота та велика кількість бібліотек та прикладів коду. Простота програмування. Низьке енергоспоживання деяких моделей;

4) *Недоліки.* Базова модель Arduino Uno має обмежену продуктивність та відсутність вбудованого Wi-Fi. Додавання модулів зв'язку збільшує вартість та розміри системи.



Рисунок 1.3 – Arduino Uno на базі мікроконтролера ATmega328P



Рисунок 1.4 – Мікроконтролер Raspberry Pi Pico

г) Raspberry Pi Pico:

1) *Виробник.* Raspberry Pi Foundation;

2) *Основні характеристики.* Мікроконтролер RP2040 з двома ядрами ARM Cortex-M0+ (до 133 МГц), велика кількість GPIO, ADC, UART, SPI, I2C, низьке енергоспоживання, компактні розміри та низька вартість,

підтримка MicroPython та C/C++. Плату мікроконтролера Raspberry Pi Pico (рис. 1.4). Для бездротового зв'язку потрібні зовнішні модулі;

3) *Переваги для систем моніторингу якості повітря.* Достатня продуктивність та велика кількість інтерфейсів. Низьке енергоспоживання;

4) *Недоліки.* Відсутність вбудованого Wi-Fi та Bluetooth.

2 РОЗРОБКА ПРОГРАМНО-АПАРАТНОЇ СИСТЕМИ АНАЛІЗУ ЯКОСТІ ПОВІТРЯ НА ОСНОВІ МІКРОКОНТРОЛЕРІВ

2.1 Теоретичні основи побудови програмно апаратних систем моніторингу

Побудова ефективних програмно-апаратних систем моніторингу, зокрема в галузі аналізу якості повітря, ґрунтується на комплексному підході, що об'єднує знання з електроніки, програмування, телекомунікацій та обробки даних. Теоретичні основи таких систем включають розуміння ключових принципів проектування, вибору компонентів, організації взаємодії між апаратними та програмними складовими, а також забезпечення надійності та масштабованості.

Основними концептуальними елементами програмно-апаратних систем моніторингу є:

- *Об'єкт моніторингу.* У нашому випадку це якість атмосферного повітря, що характеризується концентрацією різних забруднюючих речовин та фізичними параметрами (температура, вологість тощо). Розуміння характеристик об'єкта моніторингу визначає вимоги до сенсорів та частоти збору даних;
- *Сенсорна підсистема.* Включає набір сенсорів, призначених для вимірювання релевантних параметрів об'єкта моніторингу. Вибір сенсорів залежить від типу забруднювачів, необхідної точності, діапазону вимірювань та умов експлуатації. Важливими аспектами є принципи роботи сенсорів, їхні характеристики та методи калібрування для забезпечення достовірності даних;
- *Апаратна платформа обробки та комунікації.* Зазвичай представлена мікроконтролером або одноплатним комп'ютером, який здійснює збір даних з сенсорів, їхню попередню обробку (наприклад, фільтрацію, перетворення одиниць вимірювання), керує периферійними пристроями та забезпечує зв'язок з іншими елементами системи. Вибір апаратної платформи визначається необхідною

продуктивністю, кількістю інтерфейсів, наявністю засобів комунікації (дротових або бездротових) та енергоспоживанням;

– *Підсистема передачі даних.* Забезпечує транспортування зібраних даних від апаратної платформи до центрального сервера або системи зберігання та обробки. Вибір протоколів передачі даних (наприклад, UART, SPI, I2C для локальної взаємодії з сенсорами; Wi-Fi, Bluetooth, Ethernet, мобільний зв'язок для передачі на віддалений сервер) залежить від відстані, вимог до швидкості передачі та енергоефективності;

– *Програмна підсистема обробки та зберігання даних.* Включає програмне забезпечення, що функціонує на мікроконтролері (для збору та первинної обробки даних) та на сервері (для зберігання, агрегації, аналізу та візуалізації даних). Важливими аспектами є вибір мов програмування, структур даних, алгоритмів обробки та баз даних;

– *Підсистема візуалізації та інтерфейсу користувача.* Забезпечує представлення зібраних та оброблених даних у зручному для користувача форматі (графіки, діаграми, карти, таблиці). Інтерфейс користувача може бути веб-орієнтованим, мобільним додатком або іншим способом представлення інформації;

– *Підсистема живлення.* Забезпечує енергією всі компоненти системи. Вибір джерела живлення (мережа змінного струму, акумулятори, сонячні панелі) залежить від умов експлуатації та вимог до автономності.

Теоретичні основи побудови таких систем також включають важливі принципи проектування:

– *Модульність.* Розбиття системи на функціонально незалежні модулі (сенсорний, комунікаційний, обробки даних, візуалізації) полегшує розробку, тестування, налагодження та подальшу модернізацію;

– *Масштабованість.* Архітектура системи повинна передбачати можливість легкого розширення кількості датчиків та охоплення більшої території без значних змін в основній структурі;

– *Надійність та відмовостійкість.* Забезпечення стабільної роботи системи протягом тривалого часу, обробка можливих відмов окремих компонентів та мінімізація втрати даних;

– *Енергоефективність.* Оптимізація споживання енергії всіма компонентами системи, особливо при автономному живленні;

– *Безпека даних.* Забезпечення захисту зібраних даних від несанкціонованого доступу, зберігання та передачі.

Розуміння цих теоретичних основ є критично важливим для успішної розробки ефективної та надійної програмно-апаратної системи моніторингу якості повітря на основі мікроконтролерів. У наступних підрозділах ми розглянемо практичне застосування цих принципів при розробці нашої конкретної системи.

На рисунку 2.1 представлена поверхнева блок-схема розробленої програмно-апаратної системи моніторингу якості повітря. Основними функціональними блоками системи є:

– *Блок сенсорів якості повітря.* Включає набір датчиків для вимірювання різних параметрів якості повітря (наприклад, твердих частинок, концентрації газів);

– *Блок мікроконтролера.* Здійснює збір даних з блоку сенсорів, їхню попередню обробку та керує іншими компонентами системи;

– *Блок живлення*. Забезпечує електроживленням мікроконтролер та сенсори;

– *Блок модуля бездротового зв'язку.* Відповідає за передачу даних на віддалений сервер за допомогою бездротового протоколу (наприклад, Wi-Fi);

– *Сервер.* Призначений для отримання, зберігання та подальшої обробки даних, переданих з мікроконтролера;

– *Блок візуалізації.* Представляє інтерфейс (наприклад, веб-сайт або мобільний додаток) для відображення зібраних та оброблених даних про якість повітря користувачам."

Стрілками на схемі показано напрямок передачі даних та живлення між цими блоками. Дані з блоку сенсорів надходять до блоку мікроконтролера, який,

отримавши живлення від блоку живлення, передає оброблені дані через блок бездротового зв'язку на сервер. З сервера дані стають доступними для візуалізації у відповідному блоці.

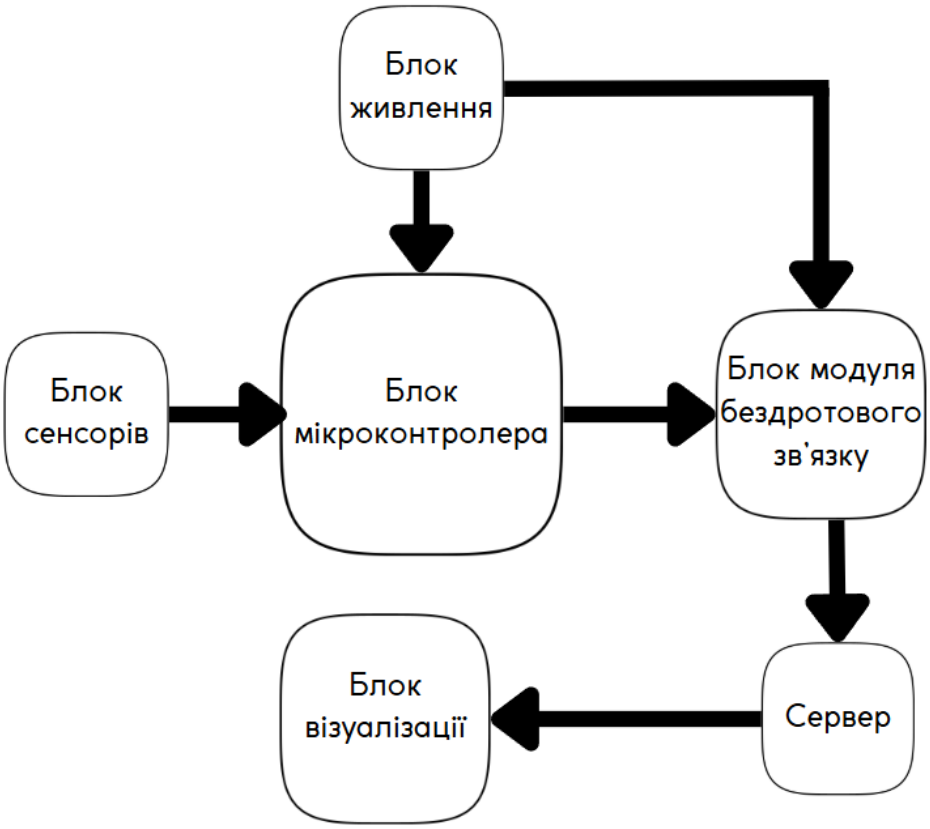


Рисунок 2.1 – «Поверхнева» блок-схема програмно-апаратної системи аналізу якості повітря

2.2 Обгрунтування вибору апаратної платформи та сенсорів

Розробка ефективної програмно-апаратної системи аналізу якості повітря вимагає ретельного вибору апаратної платформи, яка буде керувати роботою системи, обробляти дані та забезпечувати зв'язок, а також сенсорів, які будуть здійснювати безпосередні вимірювання параметрів якості повітря. У цьому

підрозділі буде обґрунтовано вибір мікроконтролера ESP32 як основної апаратної платформи та представлено обґрунтування вибору конкретних моделей сенсорів для вимірювання ключових забруднювачів.

Порівняльний аналіз та обгрунування вибору моделі ESP32. У підрозділі 1.4 було проведено огляд різних мікроконтролерів, придатних для застосування в системах моніторингу якості повітря, серед яких особливу увагу було приділено сімейству ESP32. Вибір саме ESP32 як основної апаратної платформи для розробки нашої системи зумовлений його ключовими перевагами, які вигідно вирізняють його серед інших варіантів: наявність вбудованих модулів Wi-Fi та Bluetooth, достатня продуктивність, велика кількість універсальних інтерфейсів, низьке енергоспоживання, прийнятна вартість та широка підтримка спільноти розробників. Ці характеристики роблять ESP32 ідеальним рішенням для створення бездротової, автономної та ефективної системи збору та передачі даних про якість повітря.

У рамках сімейства ESP32 існує кілька моделей, що відрізняються своїми характеристиками та функціональністю. Для визначення оптимальної моделі для нашої системи моніторингу якості повітря розглянемо порівняльну таблицю 2.1 найбільш популярних варіантів:

Таблиця 2.1 - Порівняльні характеристики популярних моделей мікроконтролерів ESP32

Модель	**ESP32-WROOM-32**	**ESP32-WROVER-B**	**ESP32-SOLO-1**	**ESP32-S3**
Процесор	Dual-core LX6	Dual-core LX6	Single-core LX7	Dual-core LX7
Тактова частота (Max)	240 МГц	240 МГц	240 МГц	240 МГц
Вбудована SRAM	520 КБ	520 КБ	320 КБ	512 КБ
Вбудована PSRAM	Ні	8 МБ	Ні	До 8 МБ
Вбудована Flash	4 МБ	4 МБ	2 МБ	8-16 МБ
Wi-Fi	2.4 GHz	2.4 GHz	2.4 GHz	2.4/5 GHz

Продовження таблиці 2.1 Порівняльні характеристики популярних моделей мікроконтролерів ESP32

Модель	**ESP32-WROOM-32**	**ESP32-WROVER-B**	**ESP32-SOLO-1**	**ESP32-S3**
Bluetooth	Classic + BLE	Classic + BLE	BLE 5.0	BLE 5.0
Кількість GPIO	До 32	До 32	До 24	До 45
ADC	12-біт, до 18	12-біт, до 18	12-біт, до 8	12-біт, до 18
DAC	2	2	Ні	2
Інтерфейси	UART, SPI, I2C	UART, SPI, I2C	UART, SPI, I2C	UART, SPI, I2C, I2S
Розмір модуля (орієнт.)	18 x 25.5 мм	18 x 31.4 мм	18 x 18 мм	18 x 25.5 мм
Основне застосування	Загальне	Висока пам'ять	Низька вартість	Безпека

На основі порівняльного аналізу для нашої системи моніторингу якості повітря найбільш оптимальною є модель ESP32-WROOM-32 .

Обґрунтуємо цей вибір:

- *Достатній обсяг пам'яті.* 520 КБ SRAM та 4 МБ Flash є достатніми для зберігання програмного коду, буферизації даних з сенсорів та реалізації необхідних алгоритмів обробки;
- *Повний набір бездротових інтерфейсів.* Вбудовані Wi-Fi (2.4 GHz) та Bluetooth (Classic + BLE) забезпечують необхідні можливості для передачі даних на сервер та потенційного локального зв'язку з іншими пристроями;
- *Достатня кількість GPIO та аналогових входів.* До 32 GPIO та до 18 аналогових входів надають гнучкість у підключенні необхідної кількості сенсорів якості повітря, а також інших периферійних компонентів;
- *Баланс між вартістю та функціональністю*. ESP32-WROOM-32 пропонує оптимальне співвідношення ціни та набору функцій, що є важливим для розробки економічно ефективної системи;

– *Широка підтримка.* Ця модель є однією з найпопулярніших у сімействі ESP32, що забезпечує велику кількість доступної документації, бібліотек та прикладів коду.

Хоча інші моделі ESP32 мають свої переваги, для нашого конкретного завдання моніторингу якості повітря можливостей ESP32-WROOM-32 є цілком достатньо, а його вартість та простота використання роблять його найбільш раціональним вибором.

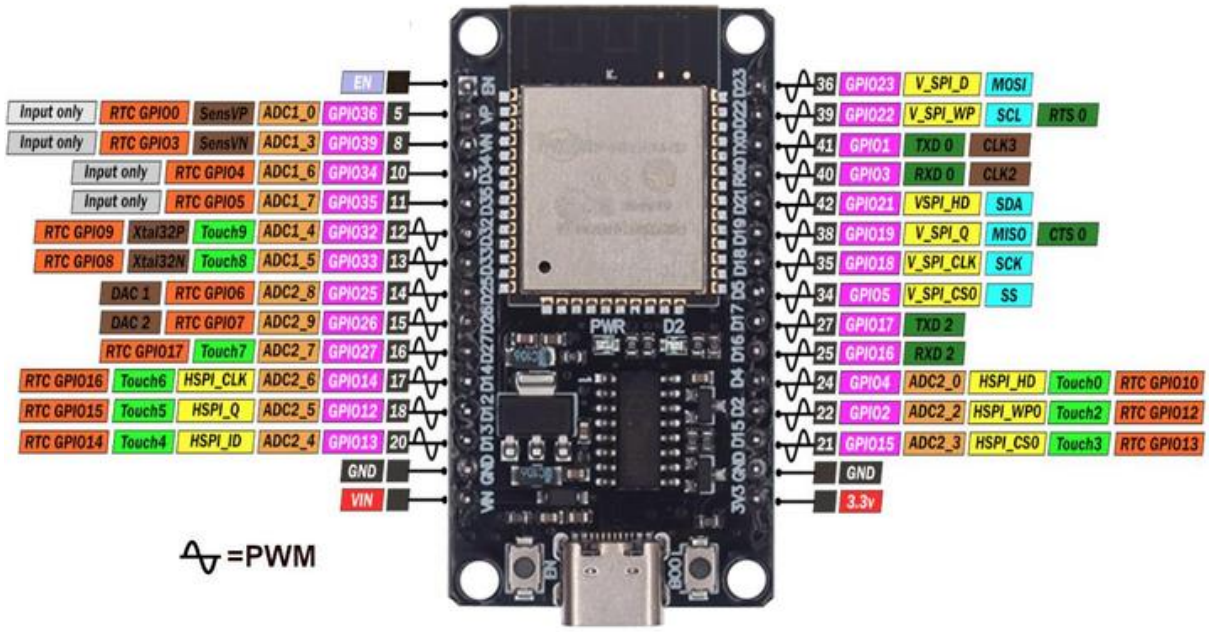


Рисунок 2.2 – Плата розробки на базі мікроконтролера ESP32-WROOM-32 з нумерацією пінів.

2.3 Вибір конкретних моделей сенсорів

Для забезпечення ефективного моніторингу якості повітря розроблена програмно-апаратна система повинна включати набір сенсорів, здатних вимірювати ключові забруднювачі. На основі критеріїв, визначених у підрозділі 2.2, було обрано наступні конкретні моделі сенсорів, а також проведено їх порівняння з можливими аналогами:

Для вимірювання твердих частинок (PM2.5 та PM10):

а) *Модель.* Plantower PMS5003. (рисунок 2.3);

б) *Обґрунтування.* Plantower PMS5003 є популярним вибором завдяки своїй балансу між ціною та продуктивністю;

в) *Аналоги.* До аналогів можна віднести Nova Fitness SDS011 та Sensirion SPS30;

г) Порівняння:

1) *SDS011* є схожим за ціною та характеристиками, але PMS5003 часто відзначається стабільнішими показаннями в умовах високої вологості;

2) *SPS30* від Sensirion є більш точним та надійним сенсором, але його вартість значно вища, що робить PMS5003 більш прийнятним для бюджетних проєктів.

д) *Вибір.* Враховуючи необхідність економічно ефективного рішення з прийнятною точністю для широкого спектра умов, було обрано Plantower PMS5003.



Рисунок 2.3 – Сенсор для вимірювання твердих частинок Plantower PMS5003

Для вимірювання монооксиду вуглецю (CO):

1) *Модель.* Металооксидний сенсор CO MQ-7 (рис. 2.4);

2) *Обґрунтування.* Для вимірювання монооксиду вуглецю було обрано металооксидний сенсор MQ-7 завдяки його простоті підключення до аналогового входу ESP32 та нижчій вартості порівняно з електрохімічними сенсорами. Хоча він має дещо нижчу точність та селективність, він є прийнятним рішенням для базового моніторингу рівня CO. Для стабільної роботи сенсор потребує попереднього прогріву;

3) *Аналоги.* Електрохімічні сенсори (наприклад, Alphasense CO-B4) та інші металооксидні сенсори (наприклад, MQ-9);

4) Порівняння:

a) *Електрохімічні сенсори* забезпечують вищу точність та селективність, але вимагають складнішого підключення (перетворення струму в напругу) та мають вищу вартість;

b) Інші *металооксидні сенсори* можуть мати різні діапазони вимірювання та чутливість, але MQ-7 є одним з найбільш поширених і має достатній діапазон для виявлення небезпечних концентрацій CO.

5) *Вибір.* Враховуючи необхідність спрощення схеми підключення на початковому етапі розробки та зниження загальної вартості системи, було обрано сенсор MQ-7. Незважаючи на його дещо нижчі характеристики порівняно з електрохімічними сенсорами, він є прийнятним для демонстрації працездатності системи та базового моніторингу CO.



Рисунок 2.4 - Металооксидний сенсор CO MQ-7

Для вимірювання летких органічних сполук (VOC):

1) *Модель.* Sensirion SGP30. (рис. 2.5);

2) *Обґрунтування.* Sensirion SGP30 був обраний за його цифровий інтерфейс, низьке енергоспоживання та відносно стабільні показання TVOC та eCO2;

3) *Аналоги.* Металооксидні сенсори серії MQ (наприклад, MQ-135), Bosch BME680 (VOC частина);

4) *Порівняння*:

a) *Сенсори серії MQ* є дешевшими, але мають нижчу точність та потребують калібрування;

b) *BME680* є хорошим комбінованим сенсором, але його VOC частина може бути менш спеціалізованою порівняно з SGP30, який розроблений саме для вимірювання VOC та eCO2;

5) *Вибір.* SGP30 був обраний завдяки своїй спеціалізації на VOC та eCO2, зручному цифровому інтерфейсу та прийнятному енергоспоживанню.



Рисунок 2.5 – Сенсор вимірювання летких органічних сполук Sensirion SGP30

Для вимірювання діоксиду вуглецю (CO2) (опціонально):

– *Модель.* MH-Z19B. (рис. 2.6);

– *Обґрунтування.* MH-Z19B є популярним NDIR сенсором CO2 завдяки своїй відносній доступності та прийнятній точності;

– *Аналоги.* Більш дорогі NDIR сенсори від SenseAir або Vaisala, а також хімічні сенсори CO2 (менш точні та надійні);

– *Порівняння.* Професійні NDIR сенсори є дорожчими, хоча й можуть мати вищу точність. Хімічні сенсори CO2 є дешевшими, але їхня точність та стабільність є значно нижчими;

– *Вибір.* MH-Z19B представляє хороший компроміс між ціною та точністю для опціонального вимірювання CO2.

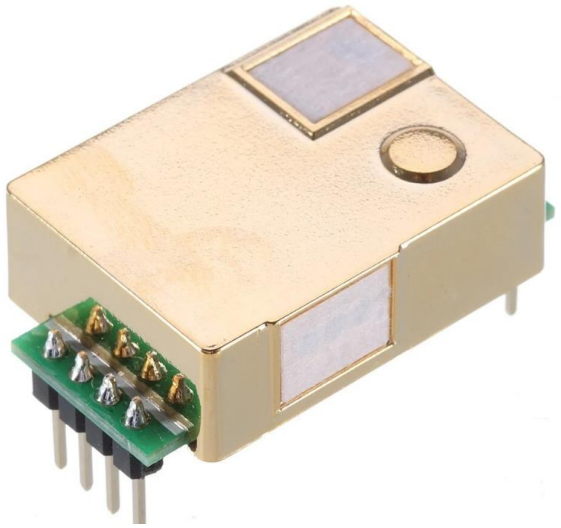


Рисунок 2.6 – Сенсор вимірювання діоксиду вуглецю (CO2) MH-Z19B

Обрані моделі сенсорів, у порівнянні з їхніми аналогами, були визначені як найбільш підходящі для розробки ефективної та інформативної системи моніторингу якості повітря на основі мікроконтролера ESP32, враховуючи вимоги до точності, вартості, енергоспоживання та доступності.

3 РОЗРОБКА ПРОГРАМНО-АПАРАТНОЇ СИСТЕМИ АНАЛІЗУ ЯКОСТІ ПОВІТРЯ НА ОСНОВІ МІКРОКОНТРОЛЕРІВ

3.1 Розробка схеми підключення сенсорів до мікроконтролера ESP32

На основі обґрунтованого вибору апаратної платформи та сенсорів у попередньому розділі, у цьому розділі буде детально розглянуто архітектуру розробленої програмно-апаратної системи аналізу якості повітря. Архітектура системи визначає взаємодію між її основними компонентами, включаючи сенсорну підсистему, мікроконтролер, підсистему передачі даних.

Ефективна робота програмно-апаратної системи аналізу якості повітря безпосередньо залежить від коректного підключення сенсорів до мікроконтролера ESP32-WROOM-32 (рис. 3.1). Для живлення системи використовується звичайний телефонний блок живлення з вихідними параметрами 5V та 2A. Від цього блоку живлення отримує енергію як сам мікроконтролер ESP32 (через пін VIN), так і датчики, що потребують напруги 5V.

У цьому підрозділі буде детально описано схему підключення обраних сенсорів (Plantower PMS5003, MQ-7, Sensirion SGP30 та опціонально MH-Z19B) до відповідних пінів ESP32, враховуючи їхні інтерфейси та потреби в живленні.

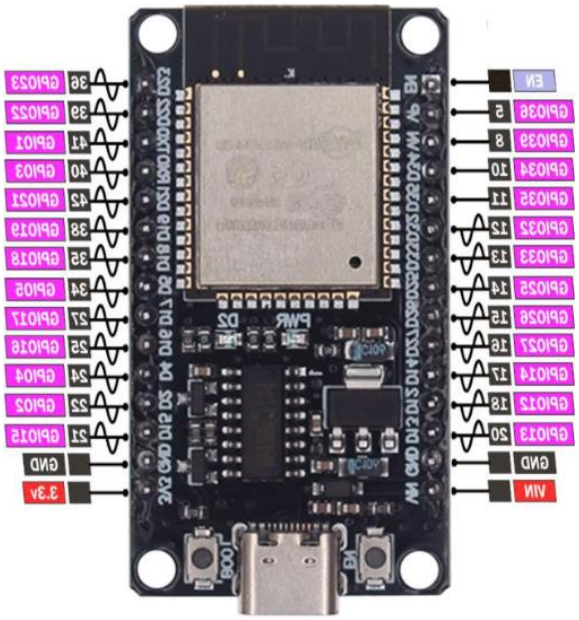


Рисунок 3.1 – Мікроконтролер ESP32-WROOM-32

Підключення Plantower PMS5003:

- TX вивід PMS5003 підключається до GPIO16 (RX2) на ESP32;
- RX вивід PMS5003 підключається до GPIO17 (TX2) на ESP32;
- VCC вивід PMS5003 підключається до 5V загального блока живлення;
- GND вивід PMS5003 підключається до GND загального блока живлення;
- Пін скидання (RESET) PMS5003 (за потреби) підключається до GPIO4 на ESP32;

Розпіновку сенсора можна побачити на рисунку 3.2.

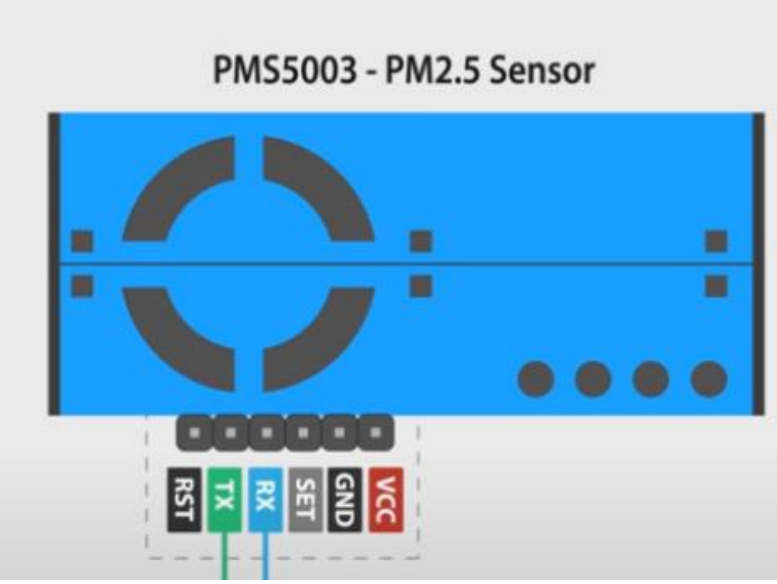


Рисунок 3.2 – Сенсор Plantower PMS5003

Підключення MQ-7:

- VCC вивід модуля MQ-7 підключається до 5V загального блока живлення;
- GND вивід модуля MQ-7 підключається до GND загального блока живлення;
- A0 (аналоговий вихід) вивід модуля MQ-7 підключається до GPIO34 (ADC1_CH6) на ESP32 для зчитування аналогового значення концентрації CO;
- D0 (цифровий вихід) вивід модуля MQ-7 може бути опціонально підключений до цифрового піна ESP32 (наприклад, GPIO2) для контролю перевищення встановленого порогу концентрації CO.

Розпіновку сенсора можна побачити на рисунку 3.3.

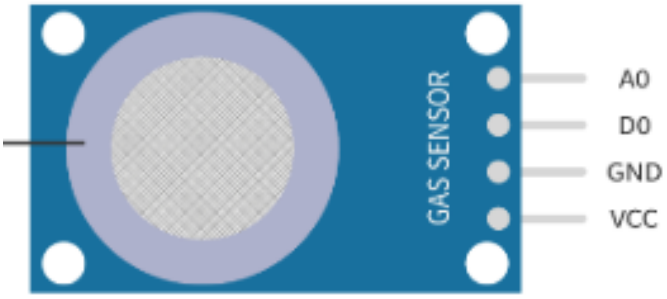


Рисунок 3.3 – Сенсор MQ-7

Підключення Sensirion SGP30:

- SDA вивід SGP30 підключається до GPIO21 (SDA) на ESP32 (I2C0);
- SCL вивід SGP30 підключається до GPIO22 (SCL) на ESP32 (I2C0);
- VCC вивід SGP30 підключається до 3.3V піна на ESP32;
- GND вивід SGP30 підключається до GND загального блока живлення.

Розпіновку сенсора можна побачити на рисунку 3.4.



Рисунок 3.4 – Сенсор SGP30

Підключення MH-Z19B (опціонально):

- TX вивід MH-Z19B підключається до GPIO18 (TX0) на ESP32 (UART0);
- RX вивід MH-Z19B підключається до GPIO19 (RX0) на ESP32 (UART0);
- VCC вивід MH-Z19B підключається до 5V загального блока живлення;
- GND вивід MH-Z19B підключається до GND загального блока живлення.

Розпіновку сенсора можна побачити на рисунку 3.5.



Рисунок 3.5 – Сенсор MH-Z19B

Описана схема підключення забезпечує взаємодію обраних сенсорів з мікроконтролером ESP32-WROOM-32 для збору необхідних даних про якість повітря, використовуючи конкретні та унікальні піни ESP32 для кожного інтерфейсу. Живлення датчиків, що потребують 5V, здійснюється від загального блока живлення, а SGP30 живиться від 3.3V виходу ESP32.

На рисунку 3.6 представлена загальна схема підключення сенсорів до мікроконтролера ESP32-WROOM-32, яка наочно ілюструє описані з'єднання та живлення компонентів від спільного блоку живлення.

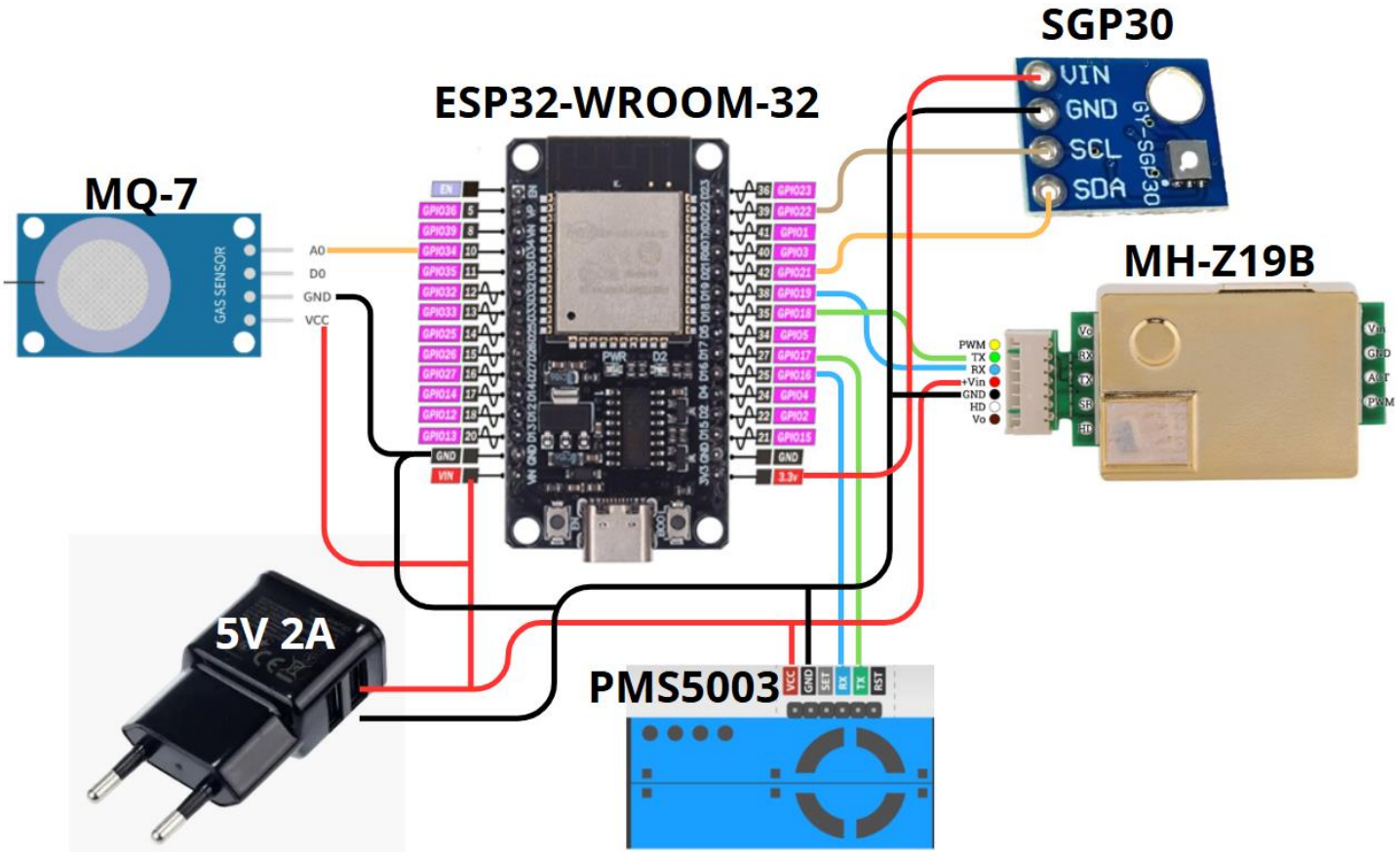


Рисунок 3.6 – Загальна схема підключення сенсорів до мікроконтролера

3.2 Опис передачі даних на сервер

Для передачі даних, зібраних сенсорами та оброблених мікроконтролером ESP32-WROOM-32, на віддалений сервер буде використано протокол HTTP *(Hypertext Transfer Protocol)*. HTTP є простим та широко застосовуваним протоколом запит-відповідь, який добре підходить для передачі даних через Інтернет. Для забезпечення безпеки передачі даних рекомендується використовувати його захищену версію – HTTPS, яка передбачає шифрування трафіку.

Формат даних для передачі буде JSON (JavaScript Object Notation). JSON є легким текстовим форматом обміну даними, який легко генерується на

мікроконтролері та обробляється на сервері. Він представляє дані у вигляді пар "ключ-значення", що добре підходить для передачі значень виміряних параметрів від сенсорів.

Структура JSON-об'єкта, що передається на сервер зображена на рисунку 3.7

```JSON
{
    "device_id": "ESP32_номер_пристрою",
    "timestamp": "YYYY-MM-DD HH:MM:SS",
    "pm25": 25.5,
    "pm10": 45.2,
    "co": 15.8,
    "tvoc": 120.3,
    "eco2": 550,
    "co2": 780
}
```

Рисунок 3.7 – Структура JSON-об'єкта, що передається на сервер.

У цьому прикладі:

- *"device_id"*. Унікальний ідентифікатор вашого ESP32 пристрою;
- *"timestamp"*. Мітка часу вимірювання даних у форматі ISO 8601;
- *"pm25", "pm10", "co", "tvoc", "eco2", "co2"*. Значення, отримані з відповідних сенсорів. Набір полів може варіюватися залежно від використовуваних сенсорів.

ESP32 буде формувати такий JSON-об'єкт з актуальними даними та відправляти його на вказану URL-адресу сервера за допомогою HTTP POST-запиту. Сервер, отримавши запит, обробляє дані та надсилає відповідь (наприклад, код статусу HTTP 200 OK у разі успішного отримання).

Частота передачі даних може бути встановлена на певний інтервал часу (наприклад, кожні 5 хвилин). ESP32 буде періодично збирати дані з сенсорів, формувати JSON-об'єкт та відправляти його на сервер.

Обробка помилок. У випадку помилки з'єднання з сервером або неотримання відповіді, ESP32 може здійснювати кілька повторних спроб відправки даних через певний інтервал часу.

Використання HTTP та JSON є простим у реалізації на ESP32 завдяки наявності відповідних бібліотек. Це забезпечує надійний спосіб передачі даних на сервер для подальшого аналізу та візуалізації.

3.3 Програмне забезпечення для збору, обробки та візуалізації даних

Програмне забезпечення розробленої програмно-апаратної системи аналізу якості повітря складається з двох основних частин: коду, що виконується на мікроконтролері ESP32 для збору та передачі даних, та програмного забезпечення на сервері для зберігання, обробки та візуалізації отриманих даних. У цьому підрозділі буде розглянуто програмне забезпечення для мікроконтролера ESP32.

Програмне забезпечення мікроконтролера ESP32. Для збору даних з підключених сенсорів (Plantower PMS5003, MQ-7, Sensirion SGP30) та їх подальшої передачі на віддалений сервер використовується код, написаний для середовища розробки Arduino IDE. Для компіляції та завантаження коду на ESP32-WROOM-32 необхідно встановити відповідну підтримку плати ESP32 в Arduino IDE, а також наступні бібліотеки через "Менеджер бібліотек":

- WiFi (вбудована);
- HTTPClient (вбудована);
- ArduinoJson від Benoît Blanchon;
- PMS5003 від FlyingFish;
- SGP30 від SparkFun або Adafruit.

Код для ESP32, який реалізує збір даних з сенсорів та їх відправку на сервер за протоколом HTTP POST у форматі JSON показано в додатку А.

Програмне забезпечення серверної частини (Django). Для прийняття даних від мікроконтролера ESP32, їхнього зберігання та візуалізації через веб-інтерфейс

використовується фреймворк Django на стороні сервера.

– *Модель даних (models.py).* Визначає структуру бази даних для зберігання даних від сенсорів. Кожен запис відповідає одному набору вимірювань у певний момент часу для конкретного пристрою. Модель бази даних на рисунку 3.8;

```
5   class SensorData(models.Model):
6       device_id = models.CharField(max_length=128)
7       timestamp = models.DateTimeField()
8
9       pm25 = models.DecimalField(max_digits=10, decimal_places=2)
10      pm10 = models.DecimalField(max_digits=10, decimal_places=2)
11
12      co = models.DecimalField(max_digits=10, decimal_places=2)
13
14      tvoc = models.DecimalField(max_digits=10, decimal_places=2)
15      eco2 = models.DecimalField(max_digits=10, decimal_places=2)
16
17      co2 = models.DecimalField(max_digits=10, decimal_places=2)
```

Рисунок 3.8 – Модель бази даних

– *Представлення (views.py).* Містить функції для обробки HTTP-запитів;

– *receive_data(request).* Приймає POST-запити з даними від ESP32 у форматі JSON. Декодує JSON та створює новий запис у базі даних SensorData. Забезпечено захист від міжсайтових запитів (CSRF) за допомогою декоратора @csrf_exempt;

Функція receive_data зображена на рисунку 3.9

```
9   @csrf_exempt
10  def receive_data(request:HttpRequest)->JsonResponse:
11      if request.method != 'POST':
12          return JsonResponse({'message':'invalid request method'}, status=400)
13
14      SensorData.objects.create(**json.loads(request.body))
15      return JsonResponse({'message':'Succes'}, status=200)
```

Рисунок 3.9 – Функція receive_data

– *serialize_obj(data: SensorData) -> dict.* Перетворює об'єкт SensorData на словник для подальшої серіалізації в JSON. Форматує мітку часу в формат ISO.

Функція serialize_obj зображена на рисунку 3.10

```
18  def serialize_obj(data: SensorData) -> dict:
19      return {
20          'device_id': data.device_id,
21          'timestamp': data.timestamp.isoformat() ,
22          'pm25': str(data.pm25),
23          'pm10': str(data.pm10),
24          'co': str(data.co),
25          'tvoc': str(data.tvoc),
26          'eco2': str(data.eco2),
27          'co2': str(data.co2),
28      }
```

Рисунок 3.10 – Функція serialize_obj

– *charts(request: HttpRequest).* Обробляє GET-запити для відображення графіків. Отримує необов'язковий параметр device для фільтрації даних за ідентифікатором пристрою. Запитує з бази даних всі дані, сортує їх за часом, серіалізує для передачі в шаблон та визначає список унікальних ідентифікаторів пристроїв для фільтрації.

Функція charts зображена на рисунку 3.11

```
31  def charts(request: HttpRequest):
32      device_id = request.GET.get('device', 'All')
33      data = SensorData.objects.order_by('timestamp')
34      data_points = []
35      devices = []
36      for o in data:
37          if o.device_id not in devices:
38              devices.append(o.device_id)
39
40          if device_id != 'All' and device_id != o.device_id:
41              continue
42          data_points.append(serialize_obj(o))
43
44      context = {'data_points': json.dumps(data_points), 'devices': sorted(devices),'selected': device_id}
45
46      return render(request, 'sensors/charts.html', context=context)
```

Рисунок 3.11 – Функція charts

Конфігурація URL (urls.py). Визначає маршрути для обробки запитів до сервера. Конфігурація зображена на фото 3.12

```
5   urlpatterns = [
6       path('admin/', admin.site.urls),
7       path('api/data', receive_data),
8       path('', charts, name='charts')
9   ]
```

Рисунок 3.12 – Конфігурація urls.py

Шаблон візуалізації (templates/sensors/charts.html). Використовує бібліотеку CanvasJS для відображення даних у вигляді графіків часових рядів. Дані передаються в шаблон як JSON-рядок (data_points). Шаблон динамічно генерує графіки для PM2.5, PM10 (на одному графіку), CO, TVOC та eCO2 (на одному графіку), а також CO2. Реалізовано фільтрацію даних за ідентифікатором пристрою за допомогою HTML-форми та Django template language, верстку можна побачити в розділі Б. Приклад візуалізації даних, отриманих за допомогою розробленої системи, наведено на рисунку 3.13.

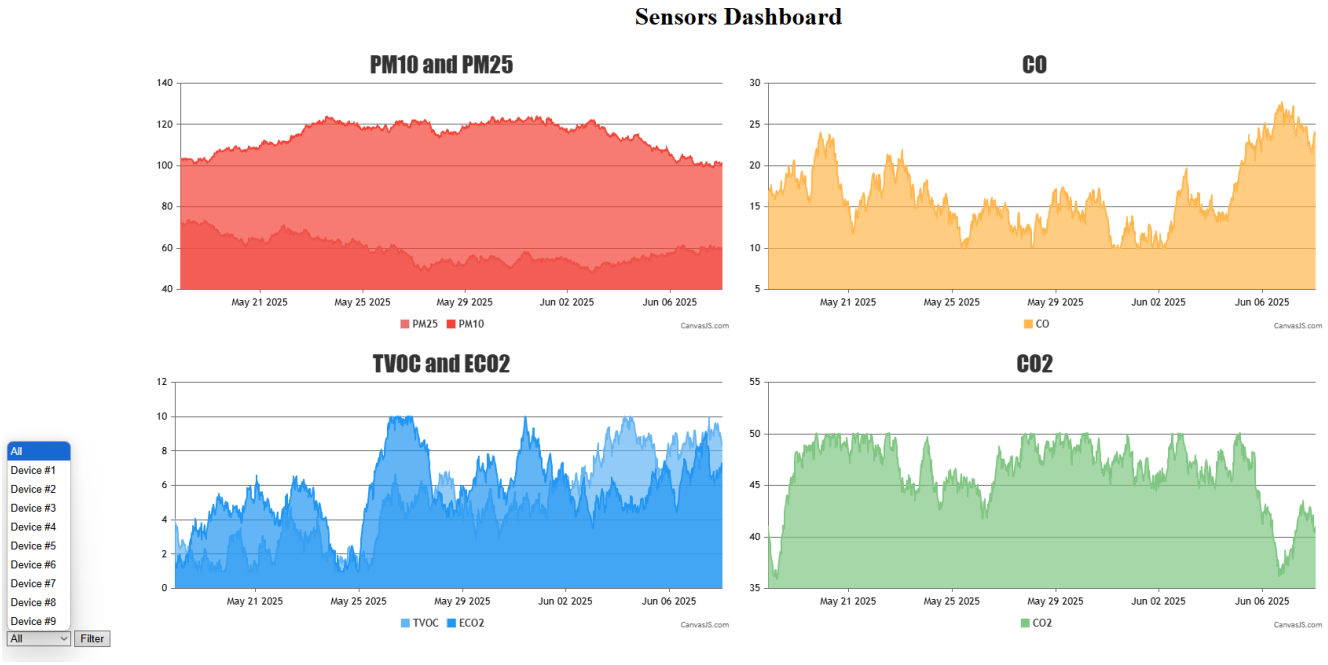


Рисунок 3.13 – Загальний вигляд візуалізації даних з сенсорі

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було проведено аналіз існуючих систем та технологій моніторингу якості повітря, розглянуто різні типи сенсорів та мікроконтролерів, що використовуються в таких системах, та розроблено теоретичну концепцію програмно-апаратної системи для аналізу якості повітря на основі мікроконтролерів ESP32.

Основні результати роботи включають:

- Детальний огляд сучасного стану проблеми забруднення повітря та обґрунтування актуальності його моніторинг;
- Аналіз існуючих систем моніторингу якості повітря (промислових, наукових та аматорських) з виділенням їхніх переваг та недоліків;
- Порівняльний аналіз сенсорів якості повітря різних типів (твердих частинок, газових, летких органічних сполук) з точки зору їхніх характеристик, принципів роботи та застосування;
- Огляд мікроконтролерів, що використовуються в системах моніторингу якості повітря, з акцентом на їхні можливості та обмеження;
- Обґрунтування вибору апаратної платформи ESP32 та конкретних моделей сенсорів для розробленої системи;
- Розробка архітектури програмно-апаратної системи, включаючи схему підключення сенсорів до мікроконтролера та опис процесу передачі даних на сервер;
- Опис програмного забезпечення для збору, обробки та візуалізації даних, включаючи вибір мови програмування, бібліотек та інструментів;
- Теоретична оцінка ефективності та потенційних результатів роботи розробленої системи на основі аналізу ключових метрик та гіпотетичних сценаріїв тестування;

Розроблена теоретична концепція системи має такі потенційні переваги:

– *Економічність*. Використання відносно недорогих сенсорів та мікроконтролера ESP32 робить систему доступною для широкого кола користувачів;

– *Гнучкість та масштабованість*. Модульна архітектура дозволяє легко додавати нові сенсори та розширювати функціональність системи;

– *Можливість створення розподілених мереж моніторингу*. Система може бути використана для створення мереж станцій моніторингу якості повітря з високою просторовою роздільною здатністю;

– *Надання даних у реальному часі*. Користувачі матимуть доступ до актуальної інформації про якість повітря, що дозволить їм приймати обґрунтовані рішення для захисту свого здоров'я.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Зима А. М. Інформаційна система оцінювання якості повітря приміщення : master's thesis. 2019. URL: http://essuir.sumdu.edu.ua/handle/123456789/75655.
2. Іваненко О. І., Гомеля М. Д., Панов Є. М. РОЗРОБКА ЗАХОДІВ ЗІ ЗНИЖЕННЯ ВИКИДІВ МОНООКСИДУ ВУГЛЕЦЮ ДИМОВИХ ГАЗІВ ПЕЧЕЙ ВИПАЛЮВАННЯ ПІД ЧАС ВИРОБНИЦТВА ЕЛЕКТРОДІВ. Scientific notes of Taurida National V.I. Vernadsky University. Series: Technical Sciences. 2020. № 4. С. 170. URL: https://doi.org/10.32838/2663-5941/2020.4/24.
3. МАРТИНЮК В., МАЛЮК О. СЕНСОРИ ТЕМПЕРАТУРИ НА БАЗІ CMOS. Herald of Khmelnytskyi National University. Technical sciences. 2024. Т. 333, № 2. С. 380–388. URL: https://doi.org/10.31891/2307-5732-2024-333-2-59.
4. ОСАДЧУК Я., ОСАДЧУК О., ОСАДЧУК В. НАДВИСОКОЧАСТОТНІ АВТОГЕНЕРАТОРНІ МІКРОЕЛЕКТРОННІ СЕНСОРИ ГАЗУ. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES. 2024. № 2. С. 143–153. URL: https://doi.org/10.31891/2219-9365-2024-78-17.
5. Павелко О., Куликовська Н., Тіменко А. Автоматизована система моніторингу якості повітря. Modeling Control and Information Technologies. 2023. № 6. С. 238–241. URL: https://doi.org/10.31713/mcit.2023.074.
6. Павелко О., Куликовська Н., Тіменко А. Автоматизована система моніторингу якості повітря. Modeling Control and Information Technologies. 2023. № 6. С. 238–241. URL: https://doi.org/10.31713/mcit.2023.074.
7. Про затвердження Порядку інформування населення про якість повітря за основними показниками з використанням індексу якості повітря в Україні : Наказ М-ва зах. довкілля та природ. ресурсів України від 24.03.2025 № 590. URL: https://zakon.rada.gov.ua/laws/show/z0536-25#Text.
8. Чижевський Т. Е., Полив'янчук А. П. Аналіз методів динамічних вимірювань вмісту твердих частинок у відпрацьованих газах дизелів : thesis. 2016. URL: http://repository.kpi.kharkov.ua/handle/KhPI-Press/46532.
9. Adrian D. R., Harmadi H. Pengaplikasian Alat Ukur Tinggi, Berat Badan, dan Penentuan Status Gizi pada Balita Berbasis ESP32 WROOM 32 melalui Telegram. Jurnal Fisika Unand. 2024. Vol. 13, no. 1. P. 82–88. URL: https://doi.org/10.25077/jfu.13.1.82-88.2024.

10. Arrahma S. A., Mukhaiyar R. Pengujian Esp32-Cam Berbasis Mikrokontroler ESP32. JTEIN: Jurnal Teknik Elektro Indonesia. 2023. Vol. 4, no. 1. P. 60–66. URL: https://doi.org/10.24036/jtein.v4i1.347.

11. Cameron N. ESP32 Microcontroller. ESP32 Formats and Communication. Berkeley, CA, 2023. С. 1–54. URL: https://doi.org/10.1007/978-1-4842-9376-8_1.

12. HTTP. Automated Data Collection with R. Chichester, UK, 2014. С. 101–148. URL: https://doi.org/10.1002/9781118834732.ch5.

13. kale J. ESP8266 NodeMCU Using Arduino IDE: Getting Start with ESP8266. Independently Published, 2018.

14. Kaur K., Kelly K. E. Laboratory evaluation of the Alphasense OPC-N3, and the Plantower PMS5003 and PMS6003 sensors. Journal of Aerosol Science. 2023. С. 106181. URL: https://doi.org/10.1016/j.jaerosci.2023.106181.

15. Kusumah H., Handayani I., Susilo P. Prototipe Monitoring Kualitas Udara Ruangan Berbasis Awan Adafruit SGP30 Air Quality Sensor. Technomedia Journal. 2018. Т. 3, № 1. С. 121–132. URL: https://doi.org/10.33050/tmj.v3i1.460.

16. Matsubara I., Itoh T., Murayama N. Significance of the Development of VOC Sensors. IEEJ Transactions on Sensors and Micromachines. 2008. Т. 128, № 4. С. 119–124. URL: https://doi.org/10.1541/ieejsmas.128.119.

17. Muqita S. G. KARAKTERISASI DAN PENGUJIAN SENSOR MQ-7 DAN MQ-136 UNTUK PENGEMBANGAN SISTEM MONITORING KONSENTRASI GAS KARBON MONOKSIDA (CO) DAN SULFUR DIOKSIDA (SO2). SEMINAR NASIONAL FISIKA 2016 UNJ. 2024. URL: https://doi.org/10.21009/03.1201.fa13.

18. Performance assessment of NOVA SDS011 low-cost PM sensor in various microenvironments / A. Božilov та ін. Environmental Monitoring and Assessment. 2022. Т. 194, № 9. URL: https://doi.org/10.1007/s10661-022-10290-7.

19. Russell D. J. ATmega328P Architecture. Introduction to Embedded Systems. Cham, 2010. С. 95–98. URL: https://doi.org/10.1007/978-3-031-79824-5_5.

20. Volders L. Raspberry Pi Pico Simplified. Lulu Press, Inc., 2021.

21. Директива 2008/50/ЄС Європейського Парламенту та Ради від 21 травня 2008 року про якість атмосферного повітря та чистіше повітря для Європи : Директива Європ. Союзу від 21.05.2008 № 2008/50/ЄС. URL: https://zakon.rada.gov.ua/laws/show/994_950#Text .

22. МАДАНІ М. ОЦІНКА ВПЛИВУ ПИЛОВИХ ВИКИДІВ У ВИРОБНИЦТВІ ГАЗОБЕТОНУ НА ЯКІСТЬ ПОВІТРЯ НОРМОВАНИХ ТЕРИТОРІЙ. *Матеріали міжнародної науково-практичної конференції "Екологія. Людина. Суспільство"*. 2024. С. 154–158. URL: https://doi.org/10.20535/ehs2710-3315.2024.304285 .

23. МАДАНІ М. ОЦІНКА ВПЛИВУ ПИЛОВИХ ВИКИДІВ У ВИРОБНИЦТВІ ГАЗОБЕТОНУ НА ЯКІСТЬ ПОВІТРЯ НОРМОВАНИХ ТЕРИТОРІЙ. *Матеріали міжнародної науково-практичної конференції "Екологія. Людина. Суспільство"*. 2024. С. 154–158. URL: https://doi.org/10.20535/ehs2710-3315.2024.304285 .

24. МАДАНІ М. ОЦІНКА ВПЛИВУ ПИЛОВИХ ВИКИДІВ У ВИРОБНИЦТВІ ГАЗОБЕТОНУ НА ЯКІСТЬ ПОВІТРЯ НОРМОВАНИХ ТЕРИТОРІЙ. *Матеріали міжнародної науково-практичної конференції "Екологія. Людина. Суспільство"*. 2024. С. 154–158. URL: https://doi.org/10.20535/ehs2710-3315.2024.304285 .

25. Навольнєв І., Максимова Н. ПОШУК ШЛЯХІВ УДОСКОНАЛЕННЯ ОЦІНКИ ВПЛИВУ ВОЄННИХ ДІЙ НА ЯКІСТЬ ПОВІТРЯ В УКРАЇНІ. *THEORETICAL AND PRACTICAL ASPECTS OF MODERN SCIENTIFIC RESEARCH*. 2025. URL: https://doi.org/10.36074/logos-24.01.2025.041 .

26. Про затвердження Порядку інформування населення про якість повітря за основними показниками з використанням індексу якості повітря в Україні : Наказ М-ва зах. довкілля та природ. ресурсів України від 24.03.2025 № 590. URL: https://zakon.rada.gov.ua/laws/show/z0536-25#Text .

27. Сейсмические датчики для систем мониторинга. *НАУКА И ТЕХНОЛОГИЧЕСКИЕ РАЗРАБОТКИ*. 2016. Т. 95, № 4. URL: https://doi.org/10.21455/std2016.4-1 .

28. Ghysels D. ESP32 programmeren voor beginners. Elektor International Media BV, 2018.

29. Grynchyshyn N. N. ЯКІСТЬ АТМОСФЕРНОГО ПОВІТРЯ ЗА ВМІСТОМ ТВЕРДИХ МІКРОЧАСТИНОК (PM 2,5) У МІСТАХ УКРАЇНИ В УМОВАХ КАРАНТИНУ ТА ВОЄННОГО СТАНУ. *Bulletin of Lviv State University of Life Safetyi*. 2023. Т. 27. С. 6–15. URL: https://doi.org/10.32447/20784643.27.2023.01 .

30. Malachynska M. J. МЕДИЧНІ ІНФОРМАЦІЙНІ СИСТЕМИ. *Medical Informatics and Engineering*. 2015. № 4. URL: https://doi.org/10.11603/mie.1996-1960.2015.4.5472 .

31. Rasberry pi for beginners. [Place of publication not identified] : Dennis Publishing, 2013.

Прошивка мікроконтролера:

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>
#include <PMS5003.h>
#include <Wire.h>
#include <SGP30.h>

const char* ssid = "ВАША_WIFI_МЕРЕЖА";
const char* password = "ВАШ_ПАРОЛЬ_WIFI";
const char* serverAddress = "http://ВАША_СЕРВЕРНА_АДРЕСА/api/data";

HardwareSerial& pmSerial = Serial2;
PMS5003 pms(pmSerial);
const int mq7Pin = 34;
SGP30 sgp30;

const char* deviceId = "ESP32_номер_пристрою";
const long interval = 300000;
unsigned long lastTime = 0;

void setup() {
    Serial.begin(115200);
    pmSerial.begin(9600, SERIAL_8N1, 16, 17);
    Wire.begin(21, 22);

    if (!sgp30.begin(Wire)) {
        Serial.println("Помилка ініціалізації SGP30");
        while (1);
    }
    Serial.print("SGP30 серійний номер: ");
    Serial.print(sgp30.serialNumber[0], HEX);
    Serial.print(sgp30.serialNumber[1], HEX);
    Serial.println(sgp30.serialNumber[2], HEX);
    sgp30.iaqInit();
    sgp30.setIAQBaseline(0x8973, 0x8AAE);

    Serial.print("Підключення до ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("Wi-Fi підключено");
    Serial.println("IP адреса: ");
    Serial.println(WiFi.localIP());
}
```

```
void loop() {
    if (millis() - lastTime > interval) {
        lastTime = millis();
        readAndSendData();
    }
    delay(1000);
}

void readAndSendData() {
    StaticJsonDocument<256> jsonDocument;
    PMS5003::AQIdata aqi = pms.readAQI();
    if (aqi.isvalid) {
        jsonDocument["pm25"] = aqi.pm25;
        jsonDocument["pm10"] = aqi.pm10;
    } else {
        Serial.println("Помилка читання PMS5003");
        jsonDocument["pm25"] = -1;
        jsonDocument["pm10"] = -1;
    }

    int mq7Value = analogRead(mq7Pin);
    jsonDocument["co"] = map(mq7Value, 0, 4095, 0, 1000);

    if (sgp30.getTVOC() && sgp30.geteCO2()) {
        jsonDocument["tvoc"] = sgp30.TVOC;
        jsonDocument["eco2"] = sgp30.eCO2;
    } else {
        Serial.println("Помилка читання SGP30");
        jsonDocument["tvoc"] = -1;
        jsonDocument["eco2"] = -1;
    }

    jsonDocument["device_id"] = deviceId;
    char timestamp[20];
    sprintf(timestamp, "%04d-%02d-%02d %02d:%02d:%02d", year(), month(),
day(), hour(), minute(), second());
    jsonDocument["timestamp"] = timestamp;
    WiFiClient client;
    HTTPClient http;
    http.begin(client, serverAddress);
    http.addHeader("Content-Type", "application/json");
    String jsonData;
    serializeJson(jsonDocument, jsonData);
    int httpResponseCode = http.POST(jsonData);
    Serial.print("HTTP Response code: ");
    Serial.println(httpResponseCode);
    Serial.print("Data sent: ");
    Serial.println(jsonData);
    http.end();
}
```

Django Project

models.py:

```
from django.db import models

class SensorData(models.Model):
    device_id = models.CharField(max_length=128)
    timestamp = models.DateTimeField()

    pm25 = models.DecimalField(max_digits=10, decimal_places=2)
    pm10 = models.DecimalField(max_digits=10, decimal_places=2)

    co = models.DecimalField(max_digits=10, decimal_places=2)

    tvoc = models.DecimalField(max_digits=10, decimal_places=2)
    eco2 = models.DecimalField(max_digits=10, decimal_places=2)

    co2 = models.DecimalField(max_digits=10, decimal_places=2)
```

views.py:

```
import json

from django.shortcuts import render
from django.http.request import HttpRequest
from django.http.response import JsonResponse
from sensors.models import SensorData
from django.views.decorators.csrf import csrf_exempt

@csrf_exempt
def receive_data(request:HttpRequest)->JsonResponse:
    if request.method != 'POST':
        return JsonResponse({'message':'invalid request method'},
status=400)

    SensorData.objects.create(**json.loads(request.body))
    return JsonResponse({'message':'Succes'}, status=200)

def serialize_obj(data: SensorData) -> dict:
    return {
        'device_id': data.device_id,
        'timestamp': data.timestamp.isoformat() ,
        'pm25': str(data.pm25),
        'pm10': str(data.pm10),
        'co': str(data.co),
        'tvoc': str(data.tvoc),
        'eco2': str(data.eco2),
        'co2': str(data.co2),
    }
```

```
def charts(request: HttpRequest):
    device_id = request.GET.get('device', 'All')
    data = SensorData.objects.order_by('timestamp')
    data_points = []
    devices = []
    for o in data:
        if o.device_id not in devices:
            devices.append(o.device_id)

        if device_id != 'All' and device_id != o.device_id:
            continue
        data_points.append(serialize_obj(o))

    context = {'data_points': json.dumps(data_points), 'devices':
sorted(devices),'selected': device_id}

    return render(request, 'sensors/charts.html', context=context)
```

urls.py:

```
from django.contrib import admin
from django.urls import path
from sensors.views import receive_data, charts

urlpatterns = [
    path('admin/', admin.site.urls),
    path('api/data', receive_data),
    path('', charts, name='charts')
]
```

Верстка візуалізації

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Sensors</title>
    <script
src="https://cdn.canvasjs.com/ga/canvasjs.min.js"></script>
</head>

<body>
    <h1 style="text-align: center;">Sensors Dashboard</h1>
    <div style="display: flex; justify-content: space-around;">
        <div
            style="display: grid; grid-template-columns: 1fr 1fr;
gap: 20px; width:80%">
            <div id="pmChart" style="width: 100%; height:
360px;"></div>
            <div id="coChart" style="width: 100%; height:
360px;"></div>
            <div id="tvocChart" style="width: 100%; height:
360px;"></div>
            <div id="co2Chart" style="width: 100%; height:
360px;"></div>
        </div>
    </div>
    <div>
        <form action="">
            <select name="device" id="deviceSelect">
                <option value="All" {% if selected == "All" %}
selected {%endif%}>
                    All</option>
                {% for device in devices %}
                <option value="{{device}}" {% if selected == device
%} selected
                    {%endif%}>{{device}}</option>
                {% endfor %}
            </select>
            <button type="submit">Filter</button>
        </form>
    </div>
</body>

</html>

<script>
```

```
        window.onload = function () {
            const data = JSON.parse(`{{ data_points | safe }}`)
            const tooltip = {
                shared: true,
            }

            const legend = {
                cursor: "pointer",
                itemclick: function (e) {
                    if (typeof (e.dataSeries.visible) === "undefined" ||
e.dataSeries.visible) {
                        e.dataSeries.visible = false;
                    } else {
                        e.dataSeries.visible = true;
                    }
                    e.chart.render();
                }
            };
            const map = (row, valueName) => {
                return { x: new Date(row['timestamp']).getTime(), y:
Number.parseFloat(row[valueName]) }
            }
            const pm25Data = [];
            const pm10Data = [];
            const coData = [];

            const tvocData = [];
            const eco2Data = [];

            const co2Data = [];

            for (let i = 0; i < data.length; i++) {
                const row = data[i]
                pm25Data.push(map(row, 'pm25'))
                pm10Data.push(map(row, 'pm10'))
                coData.push(map(row, 'co'))
                tvocData.push(map(row, 'tvoc'))
                eco2Data.push(map(row, 'eco2'))
                co2Data.push(map(row, 'co2'))
            }
            const pmChartOptions = {
                title: { text: 'PM10 and PM25' },
                data: [{
                    type: "splineArea",
                    showInLegend: "true",
                    name: "PM25",
                    color: "#e57373",
                    xValueType: "dateTime",
                    xValueFormatString: "DD MM YY HH:mm",
                    yValueFormatString: "##.##",
                    legendMarkerType: "square",
```

```
        dataPoints: pm25Data,
    }, {
        type: "splineArea",
        showInLegend: "true",
        name: "PM10",
        color: "#f44336",
        xValueType: "dateTime",
        xValueFormatString: "DD MM YY HH:mm",
        yValueFormatString: "##.##",
        legendMarkerType: "square",
        dataPoints: pm10Data
    }]
}

const coChartOptions = {
    title: { text: 'CO' },
    data: [{
        type: "splineArea",
        showInLegend: "true",
        name: "CO",
        color: "#ffb74d",
        xValueType: "dateTime",
        xValueFormatString: "DD MM YY HH:mm",
        yValueFormatString: "##.##",
        legendMarkerType: "square",
        dataPoints: coData,
    }]
}

const tvocChartOptions = {
    title: { text: 'TVOC and ECO2' },
    data: [{
        type: "splineArea",
        showInLegend: "true",
        name: "TVOC",
        color: "#64b5f6",
        xValueType: "dateTime",
        xValueFormatString: "DD MM YY HH:mm",
        yValueFormatString: "##.##",
        legendMarkerType: "square",
        dataPoints: tvocData,
    }, {
        type: "splineArea",
        showInLegend: "true",
        name: "ECO2",
        color: "#2196f3",
        xValueType: "dateTime",
        xValueFormatString: "DD MM YY HH:mm",
        yValueFormatString: "##.##",
        legendMarkerType: "square",
        dataPoints: eco2Data,
```

```
                }]
            }
            const co2ChartOptions = {
                title: { text: 'CO2' },
                data: [{
                    type: "splineArea",
                    showInLegend: "true",
                    name: "CO2",
                    color: "#81c784",
                    xValueType: "dateTime",
                    xValueFormatString: "DD MM YY HH:mm",
                    yValueFormatString: "##.##",
                    legendMarkerType: "square",
                    dataPoints: co2Data,
                }]
            }


            const pmChart = new CanvasJS.Chart('pmChart', pmChartOptions)
            const coChart = new CanvasJS.Chart('coChart', coChartOptions)
            const tvocChart = new CanvasJS.Chart('tvocChart',
tvocChartOptions)
            const co2Chart = new CanvasJS.Chart('co2Chart',
co2ChartOptions)
            pmChart.render()
            coChart.render()
            tvocChart.render()
            co2Chart.render()
        }

    </script>
```

Додаток Б

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ



Навчально-науковий інститут Інформаційних технологій
Кафедра Комп'ютерної інженерії

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

на тему: «Програмно-апаратна система аналізу якості повітря на основі мікроконтролерів»

Виконав студент групи КІД-41
Залевський Микола Олегович

Керівник кваліфікаційної роботи:
Коротков Сергій Станіславович, доцент

Київ – 2025

Мета роботи – Розробити концепцію програмно-апаратної системи для аналізу якості повітря на основі мікроконтролерів та визначити основні технічні рішення для її реалізації.

Об`єкт дослідження – Процес моніторингу та аналізу якості повітря за допомогою електронних систем.

Предмет дослідження –Архітектура програмно-апаратної системи, сенсори якості повітря, мікроконтролери та методи передачі та обробки даних для аналізу якості повітря.

Актуальність обраної теми

- **Забруднення атмосферного повітря** — це **глобальна екологічна та соціальна загроза XXI століття**. Зростання промисловості та транспорту призводить до **накопичення шкідливих речовин**, які:
- **Серйозно шкодять здоров'ю людини**, спричиняючи мільйони передчасних смертей (за даними ВООЗ) та різноманітні захворювання.
- **Негативно впливають на екосистеми**, знижують врожайність та руйнують інфраструктуру.
- **Посилюють кліматичні зміни**.
- Традиційні системи моніторингу є **дорогими та надають лише обмежені дані**. Тому існує **критична потреба** у розробці ефективних, доступних та гнучких інструментів для безперервного аналізу якості повітря.

Типи сенсорів якості повітря:

1.Сенсори твердих частинок **(Particulate Matter Sensors)**

2.Газові сенсори **(Gas Sensors):**

- Електрохімічні сенсори;
- Металооксидні (MOS) сенсори;
- Інфрачервоні (NDIR) сенсори.

3.Сенсори летких органічних сполук (VOC Sensors)

Огляд мікроконтролерів для застосування в системах моніторингу якості повітря



Рис. 1.1 ESP32



Рис. 1.2 ESP8266



Рис. 1.3 Arduino



Рис. 1.4 Raspberry Pi Pico

Основи побудови програмно апаратних систем моніторингу

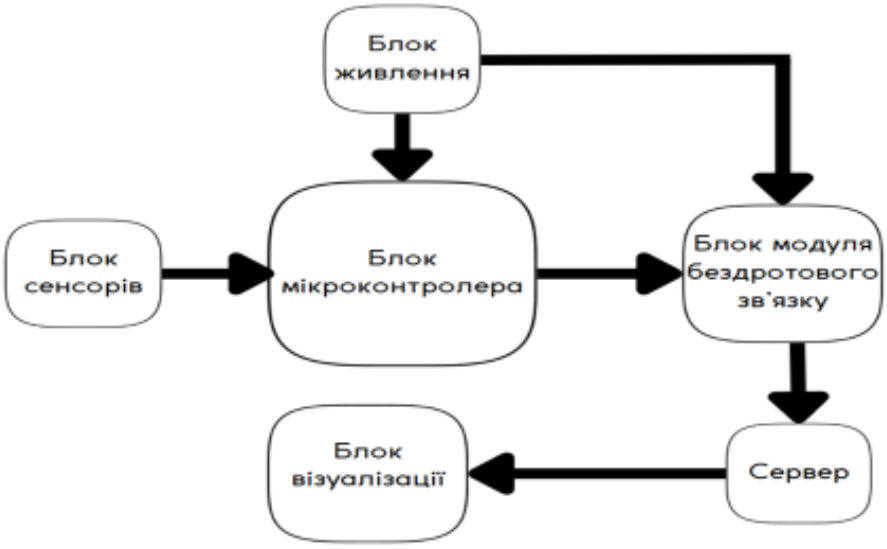


Рисунок 2.1 – «Поверхнева» блок-схема програмно-апаратної системи аналізу якості повітря

Обгрунтування вибору апаратної платформи

Модель	ESP32-WROOM-32	ESP32-WROVER-B	ESP32-SOLO-1	ESP32-S3
Процесор	Dual-core LX6	Dual-core LX6	Single-core LX7	Dual-core LX7
Тактова частота (Max)	240 МГц	240 МГц	240 МГц	240 МГц
Вбудована SRAM	520 КБ	520 КБ	320 КБ	512 КБ
Вбудована PSRAM	Ні	8 МБ	Ні	До 8 МБ
Вбудована Flash	4 МБ	4 МБ	2 МБ	8-16 МБ
Wi-Fi	2.4 GHz	2.4 GHz	2.4 GHz	2.4/5 GHz
Bluetooth	Classic + BLE	Classic + BLE	BLE 5.0	BLE 5.0
Кількість GPIO	До 32	До 32	До 24	До 45
ADC	12-біт, до 18	12-біт, до 18	12-біт, до 8	12-біт, до 18
DAC	2	2	Ні	2
Інтерфейси	UART, SPI, I2C	UART, SPI, I2C	UART, SPI, I2C	UART, SPI, I2C, I2S
Розмір модуля (орієнт.)	18 x 25.5 мм	18 x 31.4 мм	18 x 18 мм	18 x 25.5 мм
Основне застосування	Загальне	Висока пам'ять	Низька вартість	безпека

Табл 2.1 - Порівняльні характеристики популярних моделей мікроконтролерів ESP32

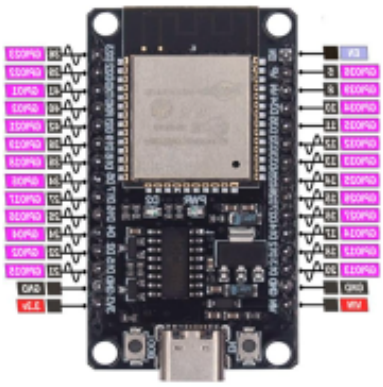


Рис 3.1 ESP32-WROOM-32

Вибір конкретних моделей сенсорів



Рис 2.3 PMS5003



Рис 2.4 CO MQ-7



Рис 2.5 SGP30



Рис 2.6 MH-Z19B

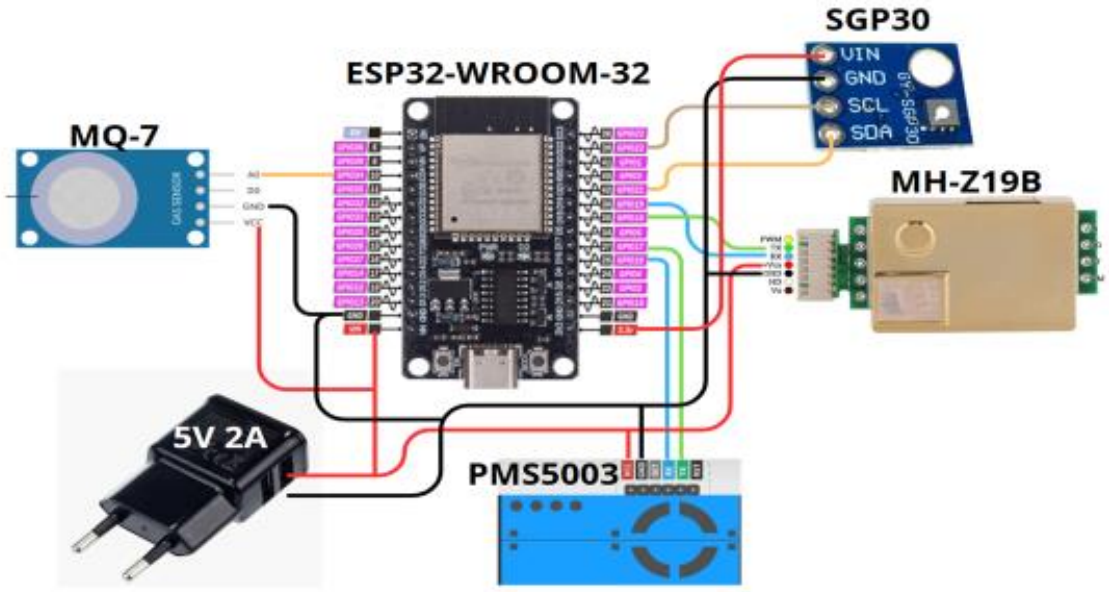


Рис 3.6 Загальна схема підключення сенсорів до мікроконтролера

Опис передачі даних на сервер

Використаємо **HTTP** для передачі даних із сенсорів на сервер, щоб було просто. Для безпеки краще **HTTPS**. Дані надсилатимемо у форматі **JSON**.

```
JSON
{
    "device_id": "ESP32_номер_пристрою",
    "timestamp": "YYYY-MM-DD HH:MM:SS",
    "pm25": 25.5,
    "pm10": 45.2,
    "co": 15.8,
    "tvoc": 120.3,
    "eco2": 550,
    "co2": 780
}
```

Рис 3.7 Структура даних, що передається на сервер

```python
class SensorData(models.Model):
    device_id = models.CharField(max_length=128)
    timestamp = models.DateTimeField()

    pm25 = models.DecimalField(max_digits=10, decimal_places=2)
    pm10 = models.DecimalField(max_digits=10, decimal_places=2)

    co = models.DecimalField(max_digits=10, decimal_places=2)

    tvoc = models.DecimalField(max_digits=10, decimal_places=2)
    eco2 = models.DecimalField(max_digits=10, decimal_places=2)

    co2 = models.DecimalField(max_digits=10, decimal_places=2)
```

Рис 3.8 – Модель бази даних

Для інтерактивної та ефективної візуалізації зібраних даних ми використовували бібліотеку Chart.js.

```python
def charts(request: HttpRequest):
    device_id = request.GET.get('device', 'All')
    data = SensorData.objects.order_by('timestamp')
    data_points = []
    devices = []
    for o in data:
        if o.device_id not in devices:
            devices.append(o.device_id)

        if device_id != 'All' and device_id != o.device_id:
            continue
        data_points.append(serialize_obj(o))

    context = {'data_points': json.dumps(data_points), 'devices': sorted(devices),'selected': device_id}

    return render(request, 'sensors/charts.html', context=context)
```

Рис 3.11 – Функція charts

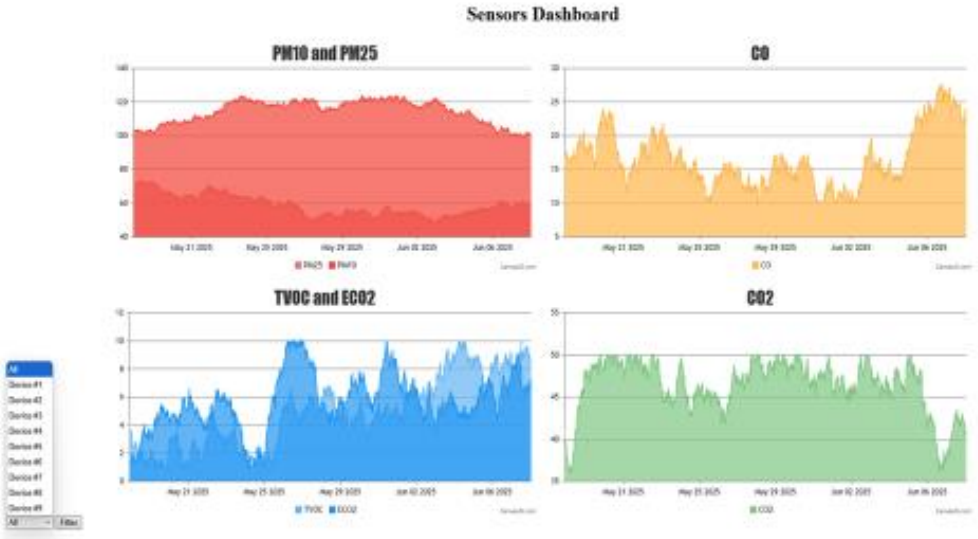


Рис 3.13 - Загальний вигляд візуалізації даних з сенсорі

Висновки

Основні результати роботи включають:

- Детальний огляд сучасного стану проблеми забруднення повітря та обґрунтування актуальності його моніторинг;
- Аналіз існуючих систем моніторингу якості повітря (промислових, наукових та аматорських) з виділенням їхніх переваг та недоліків;
- Порівняльний аналіз сенсорів якості повітря різних типів (твердих частинок, газових, летких органічних сполук) з точки зору їхніх характеристик, принципів роботи та застосування;
- Огляд мікроконтролерів, що використовуються в системах моніторингу якості повітря, з акцентом на їхні можливості та обмеження;
- Обґрунтування вибору апаратної платформи ESP32 та конкретних моделей сенсорів для розробленої системи;
- Розробка архітектури програмно-апаратної системи, включаючи схему підключення сенсорів до мікроконтролера та опис процесу передачі даних на сервер;
- Опис програмного забезпечення для збору, обробки та візуалізації даних, включаючи вибір мови програмування, бібліотек та інструментів.