Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерної інженерії та інформаційних систем

КВАЛІФІКАЦІЙНА РОБОТА

бакалавр
Освітній рівень

Кіберфізична система прогнозування опадів на основі ESP8266
Назва теми

КвРКІ 200234.21.04.87 ПЗ
Шифр

Галузь знань *12 «Інформаційні технології»*
Шифр, назва

Спеціальність *123 «Комп'ютерна інженерія»*
Шифр, назва

Освітня програма *«Комп'ютерна інженерія та програмування»*
Назва

Виконав: студент IV курсу, група КІ2-21-4 _____ Валентин КОСТЬОЛКО
Підпис Ініціали, прізвище

Керівник _____ 20.06.2025 Дмитро ДЕНИСЮК
Підпис, дата Ініціали, прізвище

Нормоконтролер _____ Тетяна КИСІЛЬ
Підпис, дата Ініціали, прізвище

До захисту допускаю:
Зав. кафедри комп'ютерної інженерії та інформаційних систем _____ Ольга ПАВЛОВА
Підпис Ініціали, прізвище

«19» червня 2025 р.

Хмельницький 2025

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

Освітній рівень БАКАЛАВР

Галузь знань 12 ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Спеціальність 123 КОМП'ЮТЕРНА ІНЖЕНЕРІЯ

Освітня програма «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ ТА ПРОГРАМУВАННЯ»

ЗАТВЕРДЖУЮ

Зав. кафедри Ольга ПАВЛОВА

" 10 " 01 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

Костьолку Валентину Руслановичу
Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) Кіберфізична система прогнозування опадів на основі ESP8266

Керівник проекту (роботи) Дмитро ДЕНИСЮК, старший викладач.
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 07.02.2025 р. № 23

2. Строк подання студентом проекту (роботи) на кафедру 01.06.2025 р.

3. Вихідні дані до проекту (роботи) Завдання на кваліфікаційну роботу

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

Вплив метрологічних даних на забезпечення умов життєдіяльності людини

Компоненти для метеостанції на основі ESP8266

Кіберфізична система прогнозування опадів на основі ESP8266

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

Функціональна схема інформаційної системи

Схема електрично принципова

Розроблена друкована плата

6. Консультанти розділів дипломного проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Тетяна КИСІЛЬ, доцент кафедри КІІС		
Антиплагіат	Андрій НІЧЕПОРУК, доцент кафедри КІІС		

7. Дата видачі завдання «_10_»_____01_____2025 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1	Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи з керівником	10.01.2025	виконано
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження	01.02.2025	виконано
3	Робота над розділом 1 – дослідження предметної області та постановка задачі	01.03.2025	виконано
4	Робота над розділом 2 – вибір компонентів для кіберфізичної системи прогнозування опадів на основі ESP8266	01.04.2025	виконано
5	Робота над розділом 3 – проєктування системи кіберфізичної системи прогнозування опадів на основі ESP8266	29.04.2025	виконано
6	Оформлення пояснювальної записки згідно вимог	25.05.2025	виконано
7	Попередній захист ВКР	26.05.2025	виконано
8	Захист ВКР на засіданні ЕК	Червень 2025 року	

Студент _____ Валентин КОСТЬОЛКО
Підпис Ініціали, прізвище

Керівник роботи _____ Дмитро ДЕНИСЮК
Підпис Ініціали, прізвище

№ рядка	формат	Позначення	Найменування	Кіл. листів	№ екз	Примітка
			Текстові документи			
1		КвРКІ 200234.21.04.87 ПЗ	Пояснювальна записка	83		
			Графічні матеріали			
2		КвРКІ 200234.21.04.87 Е8	Функціональна схема інформаційної системи	1		
3		КвРКІ 200234.21.04.72 Е8	Електрично принципова схема	1		
4		КвРКІ 200234.21.04.87 Е8	Розроблена друкована плата	1		

Зм	Арк	№ докум	Підпис	Дата	КвРКІ 200234.21.04.87 ПЗ		
Розробив		Костьолко		19.06		Літера	Аркуш	Аркушів
Перевір.		Денисюк		20.06		У	1	83
Н. контр.		Кисіль		20.06	Відомість проєкту	ХНУ, КІ2-21-4		
Затв.		Павлова		20.06				

АНОТАЦІЯ

Тема кваліфікаційної роботи: «Кіберфізична система прогнозування опадів на основі ESP8266».

Автор роботи: Валентин КОСТЬОЛКО.

Керівник роботи: Дмитро ДЕНИСЮК.

Пояснювальна записка: 83 с., 13 рис., 2 табл., 4 дод., 53 джерел.

Графічна частина: 3 креслення.

ESP8266, ARDUINO IDE, КІБЕРФІЗИЧНА СИСТЕМА, Архітектура, МОНІТОРИНГ, МЕТЕОСТАНЦІЯ.

Мета кваліфікаційної роботи полягає у розробці кіберфізичної системи для прогнозування опадів на основі мікроконтролера ESP8266, що забезпечує збір, обробку, передавання та аналіз метеорологічних даних у режимі реального часу з можливістю дистанційного доступу до результатів.

Об'єктом дослідження є процес автоматизованого моніторингу метеорологічних параметрів із застосуванням вбудованих мікропроцесорних засобів.

Предметом дослідження виступає архітектура, апаратна реалізація та програмне забезпечення автономної метеостанції, побудованої на базі ESP8266 з використанням сенсорів BMP280, BH1750, модуля вологості ґрунту та цифрового сенсора температури й вологості DHT22.

Під час проведення даного дослідження був використаний метод систематичного огляду літератури для вивчення і аналізу предметної області даного дослідження з текстових джерел інформації.

Підпис студента

30.05.2025

Дата

Зміст

ВСТУП

У сучасному світі моніторинг погодних умов відіграє надзвичайно важливу роль у багатьох сферах діяльності людини. Зміни клімату, збільшення кількості екстремальних погодних явищ, необхідність своєчасного реагування на зміни в атмосферних умовах – усе це вимагає точного та постійного контролю за погодними параметрами. Як у побуті, так і в промисловості, аграрному секторі, транспорті, будівництві та екологічному моніторингу дедалі більше зростає потреба в автоматизованих системах спостереження за кліматом.

Раніше створення метеостанцій потребувало складного обладнання та значних фінансових витрат. Проте з появою новітніх цифрових технологій, зокрема розвитку концепції Інтернету речей (IoT – Internet of Things), ситуація суттєво змінилася. IoT дозволяє створювати мережі з великої кількості пристроїв, які збирають, обробляють і передають дані через Інтернет. У рамках цієї концепції з'явилася можливість конструювати компактні, доступні за ціною, енергоефективні та прості в реалізації пристрої для моніторингу навколишнього середовища.

Одним із найефективніших інструментів для створення таких пристроїв є мікроконтролер ESP8266. Цей модуль став особливо популярним серед розробників завдяки поєднанню низької вартості, компактних розмірів, наявності вбудованого Wi-Fi-модуля та широкого спектра підтримуваних датчиків. ESP8266 може працювати як автономно, так і в складі більших систем, виконуючи функції збору, аналізу та передавання даних.

Окрім того, ESP8266 легко програмується за допомогою таких середовищ, як Arduino IDE, що відкриває широкі можливості для ентузіастів, студентів і професіоналів. Завдяки використанню цього модуля можна створити повноцінну метеостанцію, яка здатна вимірювати температуру, вологість, атмосферний тиск, рівень освітлення, наявність дощу тощо.

Таким чином, поєднання доступності, гнучкості та функціональності робить ESP8266 ідеальним вибором для розробки сучасних метеостанцій. У цьому рефераті буде розглянуто принципи побудови таких систем, основні компоненти, способи зчитування та обробки даних, а також приклади практичної реалізації.

1 ВПЛИВ МЕТРОЛОГІЧНИХ ДАНИХ НА ЗАБЕЗПЕЧЕННЯ УМОВ ЖИТТЄДІЯЛЬНОСТІ ЛЮДИНИ

В сучасному світі наявність актуальних даних про навколишнє середовище є необхідністю для комфортного життя. Починаючи від вибору одягу для майбутньої прогулянки і закінчуючи вибором оптимальної температури та вологості для комфортної роботи. Люди які часто мандрують між різними місцями України, відмічають що в залежності від міста в якому вони перебувають змінюється їхнє самопочуття та продуктивність. Причиною цього є зміна температури, вологості, тиску та інше.

Вплив метрологічних даних на забезпечення умов життєдіяльності людини**.** Метрологічні дані, отримані внаслідок контролю параметрів фізичного середовища, мають безпосередній вплив на формування безпечних та ергономічно придатних умов для людини. До ключових фізичних величин, що підлягають систематичному вимірюванню, належать температура повітря, відносна вологість, барометричний тиск, рівень освітленості, шумові навантаження та електромагнітне випромінювання. Актуальність обліку таких параметрів зумовлена необхідністю підтримки нормативних показників у сферах охорони праці, технічного регламентування та підвищення ефективності функціонування виробничих і побутових систем.

Застосування метрологічних вимірювань для моніторингу мікрокліматичних умов (відповідно до ДСТУ EN ISO 7730:2021, ГОСТ 30494-2011 тощо) дозволяє підтримувати параметри навколишнього середовища у межах, оптимальних для забезпечення фізіологічного та психоемоційного стану користувача. Наприклад, у виробничих приміщеннях рекомендоване дотримання температури у межах 20–24 °C та відносної вологості 40–60 %, що сприяє зниженню ризику перегріву, дегідратації або розвитку захворювань органів дихання.

У галузі охорони праці аналіз та облік метрологічних даних є критично важливими для виявлення і мінімізації факторів професійного ризику. Зокрема, перевищення рівня звукового тиску понад допустимі значення, встановлені

відповідно до ДСП 3.3.6.042-99, може спричинити тимчасову або хронічну втрату слуху. Аналогічно, рівень освітленості, що не відповідає вимогам ДСТУ EN 12464-1:2014, знижує візуальну продуктивність і сприяє підвищенню втомлюваності персоналу.

Метрологічне забезпечення також відіграє роль у створенні санітарно-гігієнічних умов, що впливають на якість життя. Наприклад, контроль вологості у житлових приміщеннях дозволяє зменшити ризик розвитку мікробіологічного забруднення (грибків, плісняви), що є поширеним джерелом алергенів.

Оптимізація умов праці на основі об’єктивних параметрів середовища позитивно корелює з показниками професійної продуктивності. Зниження рівня відмов у технологічних процесах та зменшення кількості браку може бути досягнуто за рахунок підтримки нормативного рівня шуму, освітленості, температурного режиму та чистоти повітря відповідно до специфікацій технічних регламентів.

Також варто зазначити, що параметри зовнішнього середовища є чинниками впливу на психоемоційний стан персоналу. Надмірні акустичні навантаження, температурний дискомфорт та неадекватне освітлення можуть бути джерелом хронічного стресу, що, у свою чергу, знижує когнітивну функціональність та загальний рівень безпеки в критичних умовах. Використання автоматизованих систем збору та аналізу метрологічних даних дозволяє виявляти подібні ризики на ранніх стадіях та здійснювати превентивні регулювальні заходи.

Використання систем контролю метеоданих в сферах промисловості, сільського господарства, транспорту, енергетики та інших галузях є ключовим для забезпечення ефективності та безпеки різних процесів.

У сільському господарстві системи контролю метеоданих використовуються для оптимізації сільськогосподарських процесів, таких як полив, внесення добрив, врожаїв та планування сівозмін. Це допомагає фермерам підвищувати врожайність та ефективність виробництва.

У транспорті системи контролю метеоданих використовуються для безпечної експлуатації транспортних засобів, планування маршрутів та уникнення небезпечних погодних умов. Це дозволяє зменшити ризики дорожніх аварій та затримок у транспортному русі.

У сфері енергетики системи контролю метеоданих використовуються для оптимізації виробництва електроенергії з відновлювальних джерел, таких як сонячна та вітрова енергія. Вони допомагають передбачати погодні умови та планувати роботу електростанцій з урахуванням прогнозованих змін у погоді.

1.1 Система контролю метеоданих в інших сферах

Застосування систем моніторингу метеорологічних параметрів у сільському господарстві**.** Інтеграція систем контролю метеорологічних даних у виробничі процеси аграрного сектору дозволяє підвищити ефективність агротехнологічного планування, знизити ризики, пов’язані з екстремальними погодними явищами, та оптимізувати використання обмежених ресурсів. Особливого значення ці системи набувають в умовах зростаючої кліматичної нестабільності, що ускладнює прогнозування агрокліматичних умов традиційними методами.

Використання метеорологічних сенсорних мереж (наприклад, IoT-станцій, що підтримують протоколи LoRaWAN або NB-IoT) дозволяє здійснювати безперервний збір даних про такі параметри як температура повітря та ґрунту, вологість, атмосферний тиск, швидкість і напрямок вітру, кількість опадів, рівень сонячної радіації. Отримані дані можуть використовуватись у складі систем підтримки прийняття рішень (DSS), що реалізують агрономічні алгоритми планування технологічних операцій, зокрема: сівби, іригації, хімічного захисту рослин, збирання врожаю.

Наприклад, за допомогою моделей транспірації та вологозабезпечення (зокрема, FAO Penman-Monteith), можна здійснювати автоматизоване регулювання поливальних систем, що забезпечує раціональне використання води та запобігає

перезволоженню кореневої зони. Аналогічно, контроль температурного режиму дозволяє вчасно активувати заходи захисту рослин від заморозків або теплового стресу.

Застосування методів аналізу часових рядів (наприклад, на основі ARIMA-моделей або методів машинного навчання RNN, LSTM) на основі історичних метеоданих дає змогу формувати коротко- та середньострокові прогнози погоди, що використовуються для динамічного коригування агрономічного плану. Такий підхід підвищує адаптивність агросистем до несприятливих погодних умов та знижує ймовірність втрат урожаю.

Крім того, метеорологічний моніторинг дозволяє оптимізувати застосування мінеральних добрив, запобігаючи як надлишковому внесенню (що призводить до втрат та забруднення навколишнього середовища), так і дефіциту, який знижує ефективність фотосинтетичних процесів. У цьому контексті погодні дані можуть інтегруватися з геопросторовою інформацією (GIS) та супутниковим дистанційним зондуванням для побудови карт агрономічної зональності.

Таким чином, впровадження систем метеорологічного моніторингу дозволяє реалізувати принципи точного землеробства, зменшуючи залежність від кліматичних факторів та підвищуючи стійкість агровиробництва до зовнішніх впливів.

Використання систем моніторингу метеорологічних параметрів у транспортній інфраструктурі**.** Застосування систем контролю метеорологічних даних у транспортній галузі є необхідним компонентом для забезпечення функціональної стійкості транспортних процесів, мінімізації ризиків дорожньо-транспортних пригод, а також підвищення ефективності логістичних операцій. Метеорологічні системи, інтегровані в транспортні інформаційно-аналітичні платформи, дозволяють в реальному часі враховувати актуальні погодні умови при формуванні маршрутів, розкладів та моделей обслуговування.

Інформація про атмосферні явища (опади, ожеледиця, туман, сильний вітер, зниження температури до критичних значень) має прямий вплив на прийняття

рішень щодо динамічного коригування маршрутів руху. Наприклад, у системах управління авіаційним та залізничним транспортом використовуються погодні прогнози з високою просторово-часовою роздільністю (зокрема, на основі моделей WRF, ICON або ECMWF), що дозволяє заздалегідь змінювати графіки перевезень або вводити тимчасові обмеження на рух.

У сфері автомобільного транспорту сенсорні вузли, розміщені вздовж дорожньої інфраструктури, забезпечують моніторинг стану дорожнього покриття (вологість, температура асфальту, наявність снігу або льоду), а також видимості та рівня освітленості. Ці дані передаються до транспортних диспетчерських центрів через бездротові мережі (наприклад, з використанням NB-IoT або LTE Cat-M1) та інтегруються в автоматизовані системи керування дорожнім рухом (АСУДД).

Крім того, погодні дані використовуються для планування технічного обслуговування транспортних засобів. Наприклад, умови знижених температур вимагають зміни регламенту обслуговування двигунів внутрішнього згоряння, перевірки акумуляторних батарей та стану шин. На основі метеоданих можливе прогнозування затримок в обслуговуванні або перенаправлення транспортних одиниць на резервні маршрути.

Застосування адаптивних транспортних систем з підтримкою аналізу метеоданих сприяє зменшенню витрат на пальне, зниженню кількості аварій, оптимізації часу доставки та підвищенню загальної якості транспортних послуг. У контексті концепції інтелектуального транспорту (ITS) погодні дані виступають одним із ключових вхідних параметрів у багатокомпонентних моделях прийняття рішень.

Використання систем моніторингу метеорологічних даних в енергетиці. Системи контролю метеорологічних параметрів є критично важливими компонентами інформаційної інфраструктури в енергетичному секторі, зокрема в контексті прогнозування попиту, оптимізації генерації, управління навантаженням і забезпечення безперервності енергопостачання. Збір і аналіз метеоданих дозволяє враховувати вплив кліматичних чинників на динаміку споживання та виробництва

електроенергії, що особливо актуально для систем з високим ступенем інтеграції відновлюваних джерел енергії (ВДЕ).

У традиційній генерації електроенергії дані про температуру навколишнього середовища, інсоляцію, швидкість вітру, хмарність та інші погодні фактори використовуються для формування добових і тижневих прогнозів навантаження на основі регресійних або гібридних моделей (наприклад, SARIMAX, XGBoost, LSTM). Такі прогнози необхідні для балансування енергосистеми, зменшення навантаження на пікові генерувальні потужності та зниження операційних витрат.

У сфері використання ВДЕ, зокрема сонячної та вітрової генерації, метеорологічні системи забезпечують дані, необхідні для побудови моделей прогнозу генерації електроенергії з урахуванням інтенсивності сонячного випромінювання, рівня хмарності, швидкості та напрямку вітру тощо. Для сонячних електростанцій (СЕС) ці дані інтегруються в системи керування на основі стандартів, таких як IEC 61724 (Photovoltaic system performance monitoring), що дозволяє адаптивно керувати інверторами, накопичувачами енергії та мережею в умовах змінної генерації.

В умовах гібридної війни, яку веде Російська Федерація проти України, особливої актуальності набуває децентралізована генерація на базі СЕС, які менш вразливі до цілеспрямованих ударів по енергетичній інфраструктурі порівняно з ТЕС і ГЕС. Встановлення сонячних електростанцій у регіонах з високим рівнем інсоляції, визначеним на основі багаторічного аналізу метеоданих, дозволяє забезпечити ефективну генерацію електроенергії при відносно низьких витратах на розгортання. В якості інструментів аналізу застосовуються дані супутникового спостереження (наприклад, NASA POWER або Copernicus Atmosphere Monitoring Service) у поєднанні з локальними сенсорними станціями.

Крім прогнозування генерації, метеорологічні системи також забезпечують критично важливу інформацію для протиаварійної автоматики та систем диспетчерського управління. За допомогою алгоритмів раннього виявлення загроз (early warning systems), заснованих на змінних погодних параметрах, можна

реалізувати запобіжне відключення або переведення споживачів у режим енергозбереження, знижуючи ризики масштабних аварій в енергосистемі.

Інтеграція метеорологічних даних у системи SCADA, EMS (Energy Management System) та DMS (Distribution Management System) дозволяє підвищити надійність функціонування енергетичних об'єктів, зменшити втрати в мережах та підвищити точність стратегічного планування енергетичного балансу з урахуванням кліматичних змін.

1.2 Методи вимірювання метеоданих

Збір метеорологічних даних передбачає фіксацію ключових параметрів атмосфери, зокрема температури повітря, відносної вологості, атмосферного тиску, швидкості та напряму вітру, кількості опадів, рівня сонячної радіації тощо. Зазначені дані є основою для побудови моделей чисельного прогнозування погоди, оцінки кліматичних змін, оптимізації аграрних процесів, забезпечення безпеки транспортних систем та обґрунтування інженерних рішень у будівництві.

Сучасні системи вимірювання атмосферних параметрів реалізуються за допомогою таких основних методів:

Наземні автоматизовані метеорологічні станції (AWS, Automated Weather Stations). Станції оснащуються датчиками, що функціонують за стандартами WMO (World Meteorological Organization), наприклад: термометрами платинового типу (Pt100) для вимірювання температури; ємнісними або психрометричними сенсорами для визначення вологості; барометрами на основі кремнієвих тензометричних елементів; анемометрами (чашковими, ультразвуковими) для фіксації вітрових параметрів; опадомірами з механізмом перекидання ковшика для вимірювання кількості опадів. Дані зазвичай передаються у цифровому форматі за протоколами RS-485, Modbus або через GSM/LoRa-зв'язок.

Радіозондові вимірювання. Метод передбачає запуск радіозондів, оснащених датчиками температури, тиску та вологості, які передають інформацію за допомогою телеметричних модулів у діапазоні VHF/UHF під час висхідного

польоту. Отримані дані дозволяють формувати вертикальні профілі атмосфери до висоти ~30 км. Стандартним є використання систем типу Vaisala RS41 або аналогічних. Результати використовуються для ініціалізації метеорологічних моделей (напр., GFS, ECMWF).

Супутникові спостереження. Орбітальні платформи (наприклад, NOAA, MetOp, Himawari) оснащуються пасивними та активними сенсорами, які здійснюють вимірювання у різних діапазонах електромагнітного спектру. Зокрема, радіометри та спектрометри (AVHRR, MODIS, IASI) реєструють температуру поверхні, структуру хмар, концентрацію водяної пари, озону тощо. Супутникові дані є критично важливими для регіонального та глобального моделювання атмосферних процесів.

Аерозондові платформи (літаки, безпілотні літальні апарати). Використовуються в умовах обмеженого доступу до цільових зон спостереження. На борту розміщуються датчики для вимірювання температури, тиску, газового складу, рівня забруднення, а також камери для візуального моніторингу. Передача даних відбувається у реальному часі через телеметричні канали. Дрон-системи типу UAV WX можуть інтегруватися з наземною інфраструктурою збору даних.

Стаціонарні метеорологічні мережі. Мережі включають як автоматизовані, так і напівавтоматизовані вимірювальні пункти. Збір даних відбувається згідно з установленими стандартами, включаючи ручні спостереження за допомогою психрометрів, барометрів, анемометрів, а також візуальні оцінки погодних умов. Дані централізовано обробляються в обчислювальних центрах для подальшого статистичного аналізу та моделювання.

Кожен із наведених методів має свої переваги та обмеження щодо точності, покриття, вартості реалізації й цільового застосування. У сучасних метеосистемах зазвичай використовується комбінація кількох методів збору даних для підвищення надійності та репрезентативності результатів.

Термопара та терморезистор як засоби вимірювання температури. Термопара – це чутливий елемент термоелектричного перетворювача, який складається з двох

ізольованих провідників із різнорідних матеріалів, з'єднаних на одному кінці. Принцип її роботи базується на термоелектричному ефекті Зеєбека, згідно з яким у місці з'єднання металів при нагріванні виникає електрорушійна сила (ЕРС), пропорційна різниці температур між гарячим і холодним кінцями провідників.

У якості термоелектродів можуть використовуватися як чисті метали – платина, мідь, залізо, так і сплави –хромель, алюмель, константан, копель, платинорідій. Завдяки своїй простоті, надійності та здатності працювати в агресивних середовищах, термопари широко застосовуються в промислових та наукових вимірюваннях. Типовий діапазон вимірювання температур для стандартних термопар становить від –50 °C до +1200 °C, залежно від типу термоелектродів.

Терморезистор (або термістор) – це температурно залежний нелінійний резистор, виготовлений з напівпровідникового матеріалу, електричний опір якого змінюється в залежності від температури. Відмінною особливістю терморезисторів є високий температурний коефіцієнт опору (ТКО), який значно перевищує значення цього параметра в металевих провідниках.

Перевагами терморезисторів є простота у використанні, висока чутливість, здатність працювати в складних кліматичних умовах, компактність, стабільність параметрів у часі та низька вартість. Залежно від знаку ТКО, терморезистори поділяються на:

- термістори – мають негативний ТКО (опір зменшується зі зростанням температури);
- позистори – мають позитивний ТКО (опір зростає зі зростанням температури).

Значення температурного коефіцієнта опору для термісторів при кімнатній температурі зазвичай становить від 0,8 % до 6 % на градус, тоді як у позисторів цей показник може сягати десятків відсотків на один градус. Завдяки цим властивостям терморезистори широко використовуються для точного вимірювання температури

навколишнього середовища, зокрема у побутовій техніці, автомобільних системах, медичних пристроях та кліматичних установках.

Вимірювання атмосферного тиску з використанням ємнісних та тензорезистивних сенсорів. Ємнісні сенсори тиску побудовані на основі кремнієвих діафрагм, що функціонують як один із електродів конденсатора. При зміні зовнішнього тиску діафрагма деформується, змінюючи відстань між нею та нерухомою опорною пластиною, що призводить до зміни ємності конденсатора. Типова конфігурація передбачає використання монолітної структури, виготовленої з монокристалічного кремнію, що забезпечує високу термомеханічну стабільність. Зміна ємності в межах до 25% від початкового значення дозволяє реалізувати пряме перетворення тиску в цифровий сигнал за допомогою відповідного аналого-цифрового перетворювача. Такі сенсори ефективні для вимірювання низького та середнього діапазонів тиску та характеризуються високою роздільною здатністю при малій споживаній потужності.

Тензорезистивні сенсори тиску реалізовані за допомогою діафрагми з інтегрованими в неї тензорезисторами, які формуються методами іонної імплантації або дифузії в монокристалічний кремній. Під впливом тиску відбувається пружна деформація діафрагми, що змінює опір тензорезистивних елементів. Ці зміни фіксуються у вигляді електричного сигналу, пропорційного прикладеному тиску. Завдяки високій модулю пружності кремнію, такі сенсори демонструють відсутність гістерезису та низьку інерційність навіть при дії високих тисків. Типовий вихідний сигнал таких сенсорів становить кілька сотень мілівольт, тому обов’язковим є використання підсилювача, що забезпечує необхідний рівень сигналу для подальшої обробки.

Основним недоліком тензорезистивних сенсорів є їх висока температурна залежність. Для компенсації температурних зсувів у практичних реалізаціях сенсорних модулів застосовують схеми температурної компенсації, наприклад, за допомогою мостових схем (типово – мостова конфігурація Вітстона) з температурно стабілізованими еталонними елементами або цифрової корекції на

рівні мікроконтролера на основі попередньо відкаліброваних температурних коефіцієнтів.

Вимірювання освітленості є невід'ємною частиною технічного аналізу ефективності штучного або природного освітлення в інтер'єрних та зовнішніх середовищах. Під освітленістю (illuminance) розуміють щільність світлового потоку на поверхні, яка вимірюється в люксах (лк, lx) відповідно до стандарту SI, де 1 лк дорівнює 1 лм/м². Контроль рівня освітленості є критичним для відповідності санітарно-гігієнічним нормам, а також для оптимізації умов зорової роботи.

Люксметри реалізують метод прямого вимірювання освітленості, що базується на фоторезистивному або фотодіодному приймачі. Вимірювальний сенсор інтегрує світловий потік, що падає на одиницю площі, з поправкою на спектральну чутливість, скориговану згідно з кривою відносної спектральної світлової ефективності людського ока V(λ)V(\lambda)V(λ), що визначена у стандарті CIE. Прилади цього типу застосовуються для швидкої оцінки локальної освітленості та калібруються відповідно до ISO 8995 (CIE S 008/E:2001).

Фотометричні методи дозволяють фіксувати просторовий розподіл інтенсивності випромінювання джерела. Використовуються фотометричні головки з вузькою діаграмою спрямованості, що дозволяє отримати кутовий розподіл світлового потоку. Результати таких вимірювань застосовуються у фотометричних аналізах (наприклад, DIALux або Relux) для моделювання світлорозподілу в приміщеннях.

Спектрофотометрія передбачає вимірювання розподілу інтенсивності світла за довжинами хвиль у діапазоні 380–780 нм. Цей метод дозволяє оцінити спектральні характеристики джерела випромінювання (наприклад, світлодіодних систем), у тому числі колірну температуру (CCT), індекс кольоропередачі (CRI), та виявити спектральні компоненти, що можуть спричиняти зоровий дискомфорт або впливати на циркадні ритми користувача. Для цього застосовуються прилади на основі дифракційної решітки або багатоканальні фотометричні датчики.

Комплексні методики поєднують вимірювання освітленості, просторового розподілу світлового потоку та спектрального складу. Прикладом є використання гониофотометрів або інтегруючих сфер для повного аналізу джерел світла. Такі підходи дозволяють оцінити відповідність освітлювального обладнання стандартам EN 12464-1 (внутрішнє освітлення робочих місць) та EN 13201 (вуличне освітлення).

1.3 Висновок до розділу

У першому розділі проведено аналіз архітектурних та функціональних засад кіберфізичних систем, призначених для моніторингу метеорологічних параметрів. Встановлено, що впровадження автоматизованих засобів вимірювання температури, вологості, атмосферного тиску та опадів є критично важливим для побудови адаптивних систем підтримки прийняття рішень у сферах агропромисловості, транспорту та енергетики.

Розглянуто методи вимірювання атмосферних параметрів із використанням наземних автоматизованих станцій, супутникового спостереження, радіозондових платформ та безпілотних літальних апаратів. Проведено технічну класифікацію сенсорних елементів для вимірювання температури (термопари, терморезистори), тиску (тензорезистивні та ємнісні сенсори), вологості, освітленості та опадів. Наведено метрологічні характеристики та протоколи передачі даних, зокрема I^2C, SPI, UART, що мають значення при розробці розподілених IoT-систем.

Проаналізовано приклади застосування метеорологічного моніторингу у практичних задачах, включаючи управління іригаційними системами, адаптивну логістику та прогнозування вироблення енергії з відновлюваних джерел. Особливу увагу приділено ролі метеоданих у забезпеченні стабільності енергетичних систем в умовах децентралізованої генерації.

На підставі аналізу визначено доцільність використання мікроконтролера ESP8266 як центрального вузла сенсорної системи. Його апаратні ресурси,

мережеві можливості та підтримка відкритих стандартів дозволяють реалізувати малогабаритні, енергоефективні системи метеомоніторингу з можливістю масштабування. Сформульовано технічні вимоги до компонентного складу майбутньої системи, що лягли в основу наступних етапів проєктування.

2 КОМПОНЕНТИ ДЛЯ МЕТЕОСТАНЦІЇ НА ОСНОВІ ESP8266

2.1 Призначення і склад системи

Портативна метеостанція призначена для вимірювання температури, атмосферного тиску та рівня освітленості навколишнього середовища. Система повинна включати три вимірювальні канали та підтримувати два режими функціонування:

- портативний режим роботи;
- стендовий режим роботи.

У портативному режимі метеостанція передає дані вимірювань через інтерфейс I2C до мікроконтролера ESP8266, який надсилає інформацію через Wi-Fi на зовнішні сервіси або ресурси, підключені до системи. У стендовому режимі результати вимірювань передаються на комп’ютер через USB-інтерфейс для подальшої обробки та аналізу. Портативна метеостанція призначена для вимірювання температури, атмосферного тиску та рівня освітленості навколишнього середовища. Вона може використовуватися як для особистих потреб, так і для комерційного використання.

2.2 Збір метеорологічних даних

Система збору метеорологічних даних, реалізована на базі мікроконтролера ESP8266, забезпечує відображення інформації у зручному та зрозумілому для користувача форматі. Завдяки передачі даних у режимі реального часу користувач отримує актуальну інформацію без затримок, що є критично важливим для оперативного реагування на зміну погодних умов.

Однією з ключових переваг є висока доступність: метеодані можуть бути переглянуті будь-яким користувачем за допомогою інтегрованих онлайн-сервісів. Це дозволяє здійснювати моніторинг стану навколишнього середовища незалежно від фізичного місця розташування користувача.

Система має гнучку архітектуру, яка дозволяє підключення додаткових сенсорів для розширення функціональних можливостей. Це відкриває перспективи масштабування та адаптації пристрою до специфічних потреб користувачів або до різних умов експлуатації.

Проєктована метеостанція повинна містити щонайменше три вимірювальні канали для збору даних про температуру, вологість, атмосферний тиск або опади, а також модуль обробки отриманої інформації. Важливою вимогою до системи є відповідність метрологічним характеристикам, що забезпечує точність і надійність вимірювань у межах встановлених норм.

Таблиця 2.1 – Метрологічні характеристики

№	Найменування параметру	Позначення	Початкове – кінцеве значення	Абсолютна похибка вимірювань
1	Температура	T	-40 - +50	±1
2	Тиск	P	500 - 1000	±1
3	Яскравість освітлення	L	0 – 10000	±1

Система повинна виконувати автоматичне вимірювання метеоданих та передавати вимірянні данні через USB або WIFI інтерфейс. Живлення системи має здійснюватися від джерела постійної напруги +5 В або 3.3 В.

2.3 Метеостанція на сонові ESP8266

Метеорологічні станції, побудовані на базі мікроконтролера ESP8266, застосовуються для вимірювання та моніторингу атмосферних параметрів у системах розподіленого середовища. До ключових технічних переваг цього рішення належать компактність, низьке енергоспоживання, вартісна ефективність

та наявність вбудованого модуля бездротового зв’язку за стандартом IEEE 802.11 b/g/n.

Апаратна архітектура метеостанції формується з окремих функціональних блоків: сенсорного блоку, модуля обробки даних, модуля живлення та комунікаційного інтерфейсу. Центральним елементом системи є мікроконтролер ESP8266, який виконує збір і первинну обробку інформації з датчиків, а також реалізує протоколи передачі даних (MQTT, HTTP, WebSocket) на зовнішні сервіси або сервери.

ESP8266 забезпечує підтримку цифрових та аналогових інтерфейсів (GPIO, I²C, SPI, 1-Wire, UART), що дозволяє підключати широкий спектр сенсорних елементів – зокрема, температурні (DS18B20), гігрометричні та барометричні (BME280, BMP180) тощо. Обчислювальних ресурсів контролера (процесор Tensilica L106 RISC, 80/160 МГц, до 4 МБ Flash-пам’яті) достатньо для реалізації алгоритмів усереднення, фільтрації та формування телеметричних повідомлень у форматі JSON.

Для практичного використання ESP8266 зазвичай інтегрується у відлагоджувальні плати типу NodeMCU v1.0 або Wemos D1 Mini, які містять USB-інтерфейс, стабілізатор напруги 3,3 В та розведені контактні колодки для прямого підключення зовнішніх компонентів. Такі плати спрощують як апаратну інтеграцію, так і процес прошивання мікроконтролера з використанням середовищ Arduino IDE або PlatformIO.

ESP8266 отримав широке застосування в системах Інтернету речей (IoT) завдяки наявності відкритої документації, великій кількості доступних бібліотек та підтримці спільнотою розробників, що спрощує розгортання автономних сенсорних вузлів для метеоспостереження.

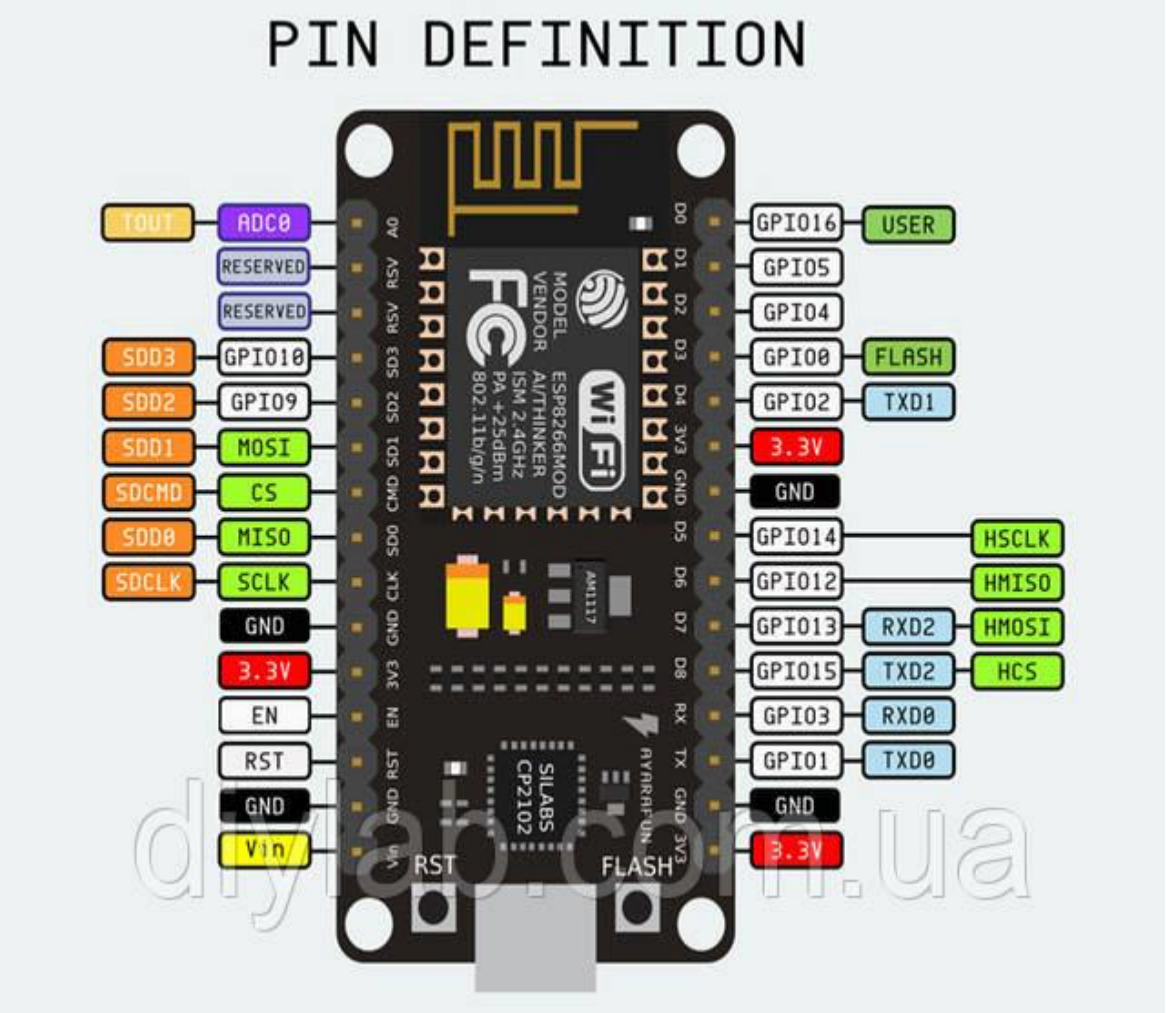


Рисунок 2.1 – Мікроконтролер ESP8266

Для вимірювання основних метеорологічних параметрів до мікроконтролера ESP8266 передбачено підключення трьох типів сенсорів, що забезпечують реєстрацію температури, вологості, атмосферного тиску та наявності опадів.

Цифрові сенсори DHT11 та DHT22 використовуються для вимірювання температури повітря та відносної вологості. Обидва сенсори працюють за власним однодротовим протоколом, передаючи дані у цифровому форматі. DHT11 має обмежений температурний діапазон (від 0 °C до +50 °C) і нижчу точність (±2 °C), тоді як DHT22 підтримує ширший діапазон температур (від –40 °C до +80 °C) і забезпечує вищу точність (до ±0,5 °C). За характеристиками вологості DHT22 також перевершує DHT11, що робить його більш придатним для застосувань, де потрібна підвищена точність вимірювань.

Для вимірювання атмосферного тиску передбачено використання сенсорів BMP180 або BME280, які обмінюються даними з мікроконтролером через інтерфейс I²C або SPI. BMP180 забезпечує вимірювання барометричного тиску в діапазоні 300–1100 гПа з точністю до ±1 гПа та додатково – температури. Сенсор BME280, окрім тиску і температури, також реєструє відносну вологість, що дозволяє зменшити кількість окремих модулів у системі. Завдяки стабільним метрологічним характеристикам обидва сенсори використовуються для визначення тенденцій зміни атмосферного тиску, що є інформативним індикатором для локального прогнозування погоди.

Для детектування наявності атмосферних опадів (дощу) використовується сенсор FC-37, принцип дії якого ґрунтується на зміні електропровідності між провідними доріжками сенсорної пластини під дією води. У разі наявності вологи опір між доріжками зменшується, що фіксується внутрішньою схемою датчика. FC-37 має два виходи: аналоговий, пропорційний рівню зволоження, та цифровий з регульованим порогом спрацьовування, реалізованим за допомогою компаратора LM393. Такий сенсор дозволяє реалізувати функцію фіксації початку опадів, а також визначати їх приблизну інтенсивність.

Запропоновані сенсори є енергоефективними, апаратно сумісними з мікроконтролерами класу ESP8266 і широко підтримуються в бібліотеках для середовищ Arduino IDE та PlatformIO. Це спрощує інтеграцію компонентів у систему та дозволяє забезпечити достовірність і повторюваність отриманих даних у рамках розроблюваної метеостанції.

Сенсори, що використовуються у системі, мають високу доступність на ринку, сумісні з мікроконтролером ESP8266 за рівнями напруги та протоколами передавання даних (I²C, цифрові GPIO), а також характеризуються мінімальними вимогами до апаратної інтеграції. Це дозволяє здійснювати безперервний моніторинг метеорологічних параметрів у режимі реального часу без необхідності використання зовнішнього комп’ютера.

Для локальної індикації зібраних даних до системи підключається дисплей. Його використання дозволяє забезпечити автономний перегляд основних параметрів без участі смартфонів або ПК. Залежно від технічних вимог до візуалізації, можуть використовуватися такі типи дисплейних модулів:

OLED-дисплей на контролері SSD1306 – графічний екран з діагоналлю 0,96 дюйма, роздільною здатністю 128×64 пікселі та інтерфейсом I²C. Перевагами є висока контрастність, низьке енергоспоживання та компактні габарити. Підтримується бібліотеками Adafruit_SSD1306, U8g2 тощо. Може використовуватись для виведення значень температури, тиску, вологості у вигляді тексту або простих графіків.

TFT-дисплей – кольоровий дисплей з діагоналлю від 1,8 до 2,4 дюйма, роздільною здатністю до 320×240 пікселів, що використовує SPI для обміну даними. На відміну від OLED, TFT-дисплей дає змогу виводити повноцінну графіку (наприклад, гістограми, тренди, піктограми стану системи), що покращує інтерпретацію інформації в реальному часі. Прикладом є дисплеї на контролерах ILI9341 або ST7735.

Енергозабезпечення метеостанції може реалізовуватись у двох режимах: стаціонарному та автономному. У стаціонарному варіанті живлення подається від зовнішнього адаптера постійного струму з номіналом 5 В або 3,3 В через стабілізатор напруги (наприклад, AMS1117). Для автономної роботи передбачено використання літієвих акумуляторів типу Li-Ion (3,7 В, ємність 18650) або Li-Po. У разі використання сонячної панелі для заряджання акумулятора додається контролер живлення (наприклад, TP4056 з захистом від перенапруги та надрозряду).

Оскільки мікроконтролер ESP8266 є чутливим до нестабільної напруги (робочий діапазон живлення 3,0–3,6 В), особливу увагу слід приділяти якості джерела живлення. Коливання напруги живлення можуть призводити до порушень у роботі модуля Wi-Fi або до некоректної обробки даних із сенсорів, тому

стабілізація подачі живлення є критичною умовою для надійного функціонування системи.

ESP8266 можна програмувати за допомогою середовища IDE Arduino або інших платформ, таких як PlatformIO. Використання відповідних бібліотек для роботи з датчиками та дисплеями може значно спростити написання коду. ESP8266 також можна інтегрувати з веб-сервісами:

– OpenWeatherMap – це сервіс, який дозволяє отримувати прогноз погоди через Інтернет.
– ThingSpeak – це хмарна платформа для зберігання, аналізу та візуалізації даних з пристроїв IoT.

Таким чином, архітектура метеостанції на основі ESP8266 є модульною та масштабованою. Залежно від ваших потреб ви можете легко додавати нові функції, змінювати датчики, збирати статистику, впроваджувати прогнози погоди тощо.

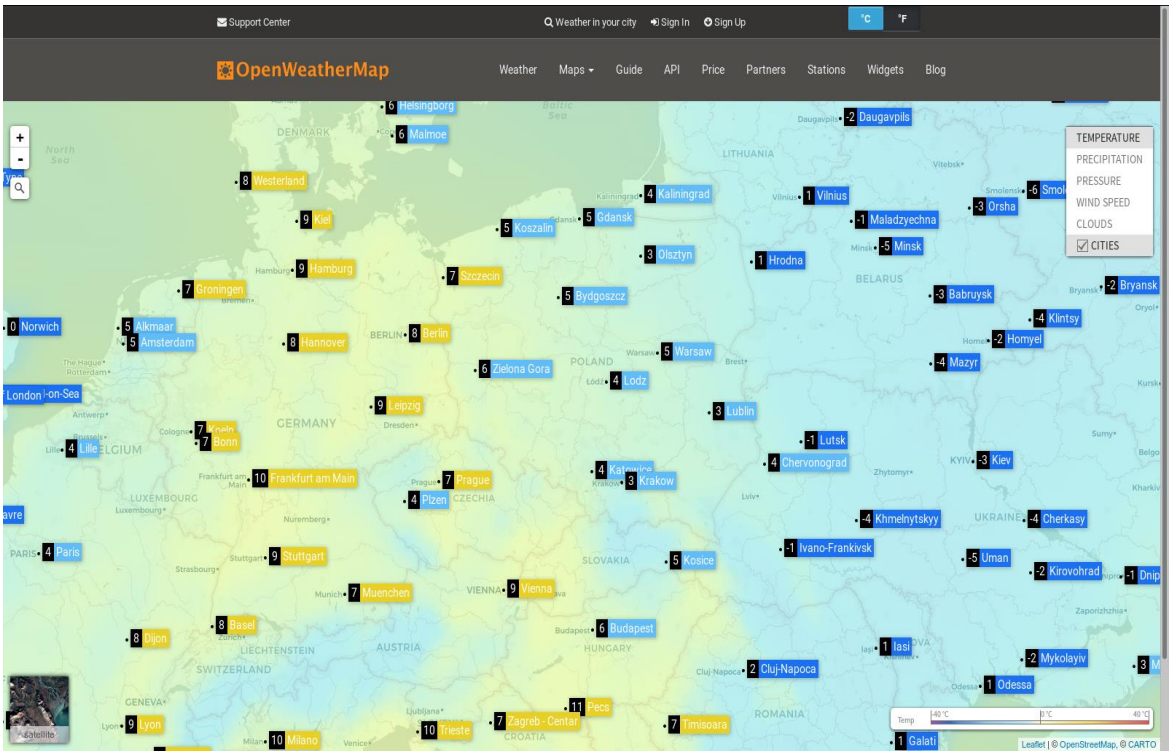


Рисунок 2.2 - Сервіс OpenWeatherMap

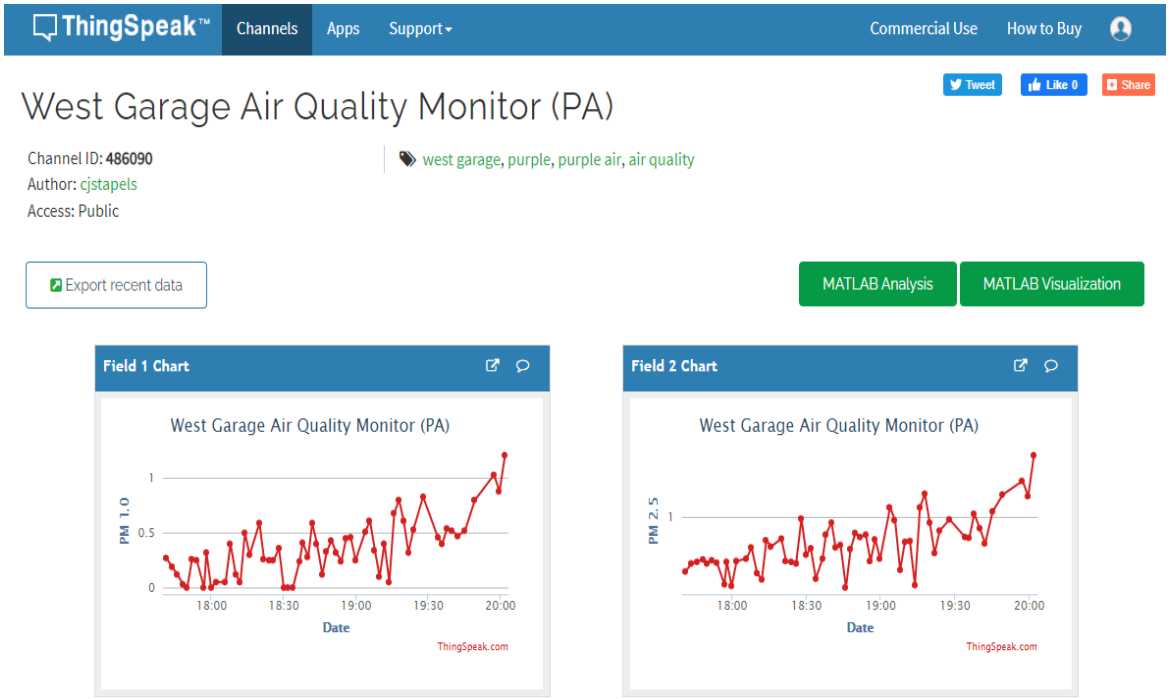


Рисунок 2.3 - Хмарна платформа ThingSpeak

2.4 Аналіз сенсорних модулів для метеостанції

Ефективна робота метеостанції ESP8266 залежить від правильного підключення датчиків, які вимірюють ключові параметри навколишнього середовища. Для отримання потрібних даних потрібно не тільки фізично підключити датчик до мікроконтролера, а й налаштувати програмну частину системи (з відповідними бібліотеками та функціями обробки сигналів). У цьому розділі описано, як підключити та налаштувати три основні датчики: датчик температури та вологості (DHT11/DHT22), датчик тиску повітря (BMP180/BME280) і датчик дощу (FC-37).

Датчики DHT11 та DHT22 є одними з найпоширеніших пристроїв для вимірювання температури та відносної вологості повітря у недорогих мікропроцесорних системах. Вони мають цифровий вихідний інтерфейс, що дозволяє передавати виміряні значення без необхідності додаткового аналого-цифрового перетворення. Обидва сенсори підключаються до мікроконтролера

ESP8266 через один загальний цифровий вивід, що значно спрощує апаратну реалізацію.

Основна відмінність між цими двома моделями полягає в точності та робочому діапазоні. DHT11 підтримує вимірювання температури в межах від 0 до +50 °C та вологості до 80 % RH з помірною точністю. Натомість DHT22 дозволяє здійснювати вимірювання у значно ширшому температурному діапазоні (від –40 до +80 °C) і підтримує до 100 % вологості, забезпечуючи водночас вищу точність.

Для підключення сенсора до мікроконтролера необхідно з’єднати вивід живлення VCC з лінією живлення ESP8266 (3,3 В або 5 В залежно від використовуваної плати), вивід GND – із загальною "землею", а лінію даних DATA – з одним із вільних цифрових GPIO-виводів. Для забезпечення стабільної роботи інтерфейсу між живленням та лінією даних рекомендується встановлювати підтягувальний резистор номіналом 10 кОм.

У програмній частині системи для взаємодії з цими сенсорами використовується бібліотека DHT.h, яка доступна в офіційному менеджері бібліотек середовища розробки Arduino IDE. Після її підключення необхідно оголосити відповідний об’єкт для роботи із сенсором та вказати номер GPIO-виводу, до якого підключено пристрій. Подальше зчитування даних здійснюється через стандартні функції бібліотеки, які дозволяють отримувати значення температури та вологості в зручному для обробки вигляді.

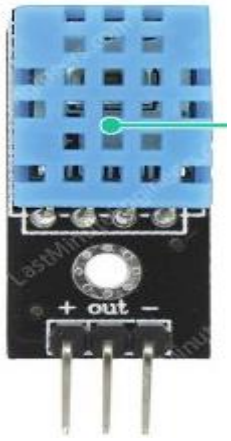


Рисунок 2.4 – Датчики DHT11 [33]

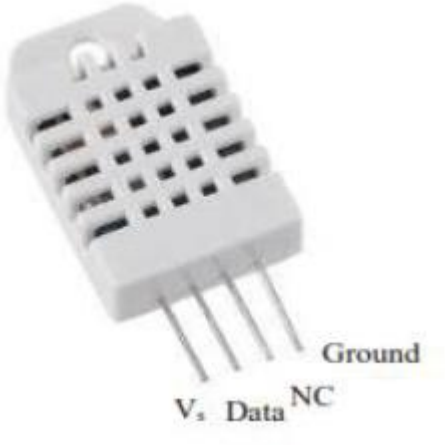


Рисунок 2.5 – Датчики DHT22 [16]

Датчики BMP180 та BME280 використовуються для вимірювання атмосферного тиску та температури повітря, що є важливими параметрами для побудови метеорологічних систем. Обидва сенсори належать до категорії високоточного обладнання для цифрових вимірювань. Основною перевагою BME280 порівняно з BMP180 є наявність вбудованого модуля вимірювання відносної вологості, що робить його більш універсальним рішенням для збору метеорологічних даних.

Обидва датчики підтримують два цифрові інтерфейси – I2C і SPI – однак у більшості випадків використовується інтерфейс I2C, оскільки він потребує менше з'єднань і значно спрощує процес підключення до мікроконтролера ESP8266. Для реалізації обміну даними за інтерфейсом I2C необхідно підключити виводи живлення (VCC) і "землі" (GND) відповідно до контактів ESP8266, а також передбачити з’єднання сигнальних ліній: вивід SCL (Serial Clock) – до одного з цифрових виводів мікроконтролера (наприклад, GPIO5), і SDA (Serial Data) – до іншого (наприклад, GPIO4).

Для забезпечення коректного функціонування сенсорів необхідно використати відповідне програмне забезпечення. Для BMP180 застосовується бібліотека Adafruit_BMP085.h, а для BME280 – Adafruit_BME280.h. Обидві бібліотеки вимагають наявності допоміжної бібліотеки Adafruit_Sensor.h, яка забезпечує уніфікований підхід до збору та обробки даних з різних типів сенсорів.

Після підключення цих бібліотек у програмному середовищі (наприклад, Arduino IDE) можна ініціалізувати обраний сенсор і зчитувати з нього значення тиску, температури, а за потреби – й вологості. Це дозволяє ефективно інтегрувати дані в загальну систему моніторингу метеоумов у режимі реального часу.

.

Рисунок 2.6 - Датчик BMP180 [14]

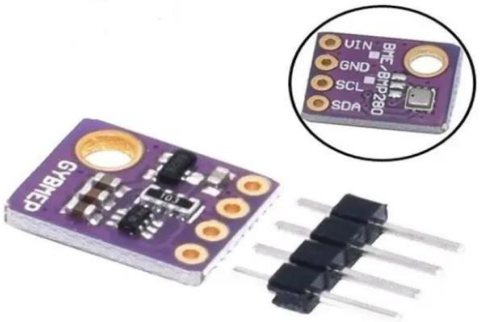


Рисунок 2.7 - Датчик BME280 [35]

Датчик дощу FC-37 використовується для виявлення опадів шляхом визначення наявності води на його поверхні. Принцип його роботи ґрунтується на зміні електричного опору між провідними доріжками на спеціальній підкладці.

Коли на цю поверхню потрапляють краплі води, провідність між доріжками збільшується, що викликає зміну сигналу, який передається до мікроконтролера.

До складу модуля входить сама сенсорна пластина та підсилювальний модуль, який забезпечує обробку сигналів і дозволяє зчитування як аналогових, так і цифрових значень. Цифровий вихід реагує на настання або відсутність факту змочування поверхні, тоді як аналоговий вихід може використовуватись для оцінки інтенсивності вологості або кількості опадів.

Підключення модуля до мікроконтролера ESP8266 відбувається через стандартні контакти живлення та сигнальні лінії. Вивід живлення (VCC) підключається до джерела 3,3 В, а вивід заземлення (GND) – до загальної "землі". Цифровий вихід DO може бути з’єднаний з будь-яким вільним цифровим піном ESP8266, наприклад, D5, тоді як аналоговий вихід AO підключається до єдиного аналогового входу мікроконтролера – A0.

З програмної точки зору, цифровий сигнал дозволяє зчитувати логічний рівень, що сигналізує про наявність чи відсутність води. Для кращої гнучкості використання датчика можна налаштувати порогове значення чутливості, яке визначає, при якому рівні вологості буде згенеровано відповідний цифровий сигнал. Це дозволяє адаптувати роботу сенсора до конкретних умов середовища та уникнути хибних спрацьовувань через конденсат чи незначну вологість.

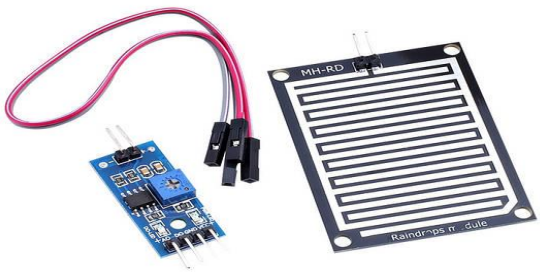


Рисунок 2.8 - Датчик FC-37 [39]

2.5 Засоби візуалізації результатів вимірювання

Візуалізація є невід'ємною складовою сучасних метеостанцій, оскільки вона забезпечує наочне представлення метеорологічних даних, що збираються пристроєм у режимі реального часу. Наявність вбудованого дисплея дозволяє користувачу оперативно отримувати інформацію про поточну температуру, вологість, атмосферний тиск або наявність опадів без необхідності підключення до зовнішніх пристроїв, таких як смартфон чи комп'ютер.

Завдяки локальному відображенню інформації, система стає більш автономною та зручною у використанні. У проєктах на базі мікроконтролера ESP8266 переважно використовуються два типи дисплеїв: OLED та TFT. Кожен з них має свої функціональні можливості, що визначають їх доцільність у конкретних умовах застосування. Наприклад, OLED-дисплеї забезпечують високу контрастність та мале енергоспоживання, тоді як TFT-дисплеї дають змогу реалізувати кольорову графіку й складніші візуальні елементи.

Вибір дисплея залежить від вимог до розміру екрана, якості зображення, енергоспоживання та умов експлуатації.

OLED-дисплеї (Organic Light-Emitting Diode) стали одним із найпопулярніших рішень для відображення інформації в компактних електронних пристроях, зокрема в портативних метеостанціях на базі ESP8266. Завдяки технології самосвітних пікселів, ці дисплеї не потребують додаткового підсвічування, що суттєво знижує енергоспоживання.

Найчастіше в таких проєктах використовується дисплей на основі контролера SSD1306. Він забезпечує достатню для відображення числових та графічних даних роздільну здатність і чудову видимість навіть при слабкому освітленні. Компактні розміри дозволяють вбудовувати OLED-дисплей у невеликі корпуси без шкоди для ергономіки пристрою.

Практичне використання OLED-дисплея полягає у виведенні поточних метеоданих – температури, вологості, атмосферного тиску, а також простих

графічних елементів, таких як піктограми погоди чи стрілки тенденції. Завдяки цьому користувач може з першого погляду оцінити поточні умови без потреби аналізувати числові значення.

Втім, OLED-дисплеї мають і певні обмеження. Зокрема, вони не підтримують кольорове відображення, що зменшує візуальну привабливість та можливості деталізації графіки. Крім того, обмежена площа екрану не завжди дозволяє одночасно показати всі необхідні параметри, особливо якщо використовується великий шрифт чи додаткові символи.

Незважаючи на ці обмеження, OLED залишається ефективним рішенням для метеостанцій, де важлива компактність, енергоефективність та чіткість відображення.

Рисунок 2.9 - OLED-дисплей SSD1306 [28]

TFT-дисплеї (Thin Film Transistor) відкривають значно ширші можливості для візуалізації даних у метеостанціях завдяки своїм характеристикам. Вони підтримують відображення кольорових зображень, мають більшу площу екрана порівняно з OLED і дозволяють створювати більш складний графічний інтерфейс.

Кольорове зображення є головною перевагою TFT-дисплеїв. Це дає змогу не лише виводити числові показники, а й будувати барвисті графіки, використовувати кольори для позначення різних погодних умов і навіть реалізовувати прості анімації, що покращують зручність сприйняття інформації.

Вища роздільна здатність дозволяє розміщувати більше даних на одному екрані або деталізувати графіки. Деякі моделі також мають сенсорний екран, що

дає можливість створити простий інтерфейс користувача, наприклад, для перемикання між сторінками або зміни налаштувань.

На практиці TFT-дисплей у метеостанції може виконувати функцію дашборду з декількома інформаційними блоками. Наприклад, одна сторінка може показувати температуру та вологість, інша – графік зміни температури за останні години, ще одна – прогноз чи загальний стан погоди. Перемикання між екранами може відбуватись автоматично або за допомогою сенсора чи кнопки.

Проте такі дисплеї мають і свої недоліки. Вони споживають більше енергії, тому менш придатні для автономних пристроїв, які працюють від акумуляторів. Крім того, їх підключення потребує більше виводів та ресурсів мікроконтролера, що може ускладнити реалізацію в обмеженому за обчислювальними можливостями середовищі, як-от ESP8266.

Незважаючи на це, TFT-дисплеї – чудовий вибір для стаціонарних або енергозабезпечених метеостанцій, де ключову роль відіграє повноцінна візуалізація та зручність інтерфейсу.

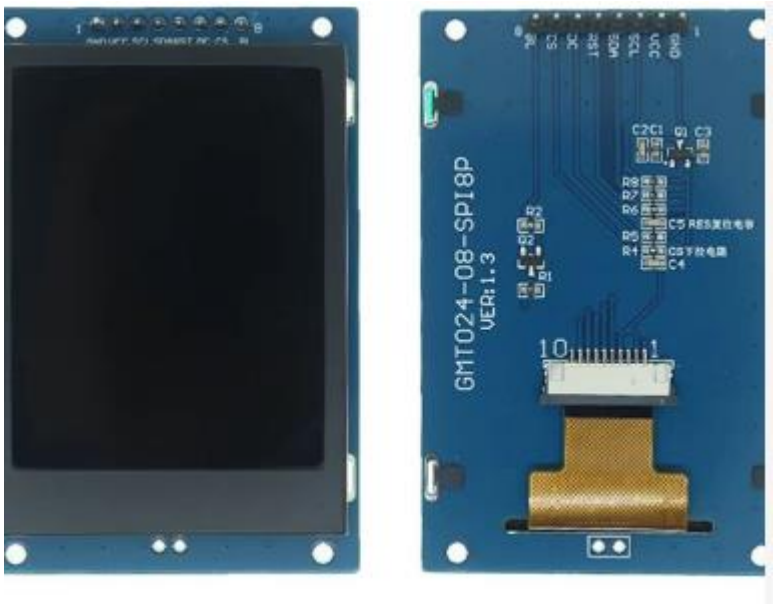


Рисунок 2.10 - TFT-дисплей [50]

Порівняння OLED vs TFT

Таблиця 2.2 – Характеристика дисплеїв

Характеристика	**OLED SSD1306**	**TFT-дисплей**
Тип зображення	Монохромне	Кольорове
Розмір дисплея	Невеликий (0.96", 1.3")	Більший (1.8", 2.4", 2.8" і більше)
Споживання енергії	Низьке	Високе
Інтерфейс підключення	I2C (2 лінії)	SPI (4–6 ліній)
Відображення графіки	Проста, символічна	Повноцінна, графічна
Ідеальне застосування	Автономні станції	Стаціонарні системи

Однією з найсильніших сторін модуля ESP8266 є його здатність діяти як повноцінний мережевий пристрій, що поєднує апаратну та програмну частину для збору, обробки і передачі метеорологічних даних. Його інтегрований Wi-Fi-модуль дозволяє напряму взаємодіяти з Інтернетом без потреби у додаткових шлюзах чи маршрутизаторах.

ESP8266 легко інтегрується з великою кількістю сенсорів – як цифрових, так і аналогових. Пристрої на його основі можуть регулярно зчитувати значення температури, вологості, тиску, рівня освітленості або наявності дощу. Опитування датчиків зазвичай виконується в циклі з певним інтервалом, наприклад, кожні 30 або 60 секунд. Отримані значення зберігаються у змінних для подальшої обробки або виводу на екран.

Найпоширеніший спосіб комунікації з вебсервісами – використання HTTP-запитів. За допомогою бібліотек ESP8266WiFi.h та HTTPClient.h, мікроконтролер формує GET або POST-запити для передачі зібраної інформації на сервер або для отримання даних (наприклад, прогнозу погоди). Такий підхід простий і добре підходить для взаємодії з REST API.

MQTT – ефективний протокол для обміну повідомленнями у мережах пристроїв IoT. ESP8266 може публікувати дані на MQTT-брокер або підписуватись на теми для отримання команд. Завдяки цьому можливе створення розподілених систем, у яких багато пристроїв обмінюються даними в режимі реального часу. MQTT підходить для проектів з мінімальним споживанням енергії та мережевим навантаженням.

OpenWeatherMap надає погодні дані у форматі JSON, які можна використовувати для порівняння з локальними показниками, перевірки точності сенсорів або відображення додаткової інформації, такої як швидкість вітру чи хмарність. За допомогою бібліотеки ArduinoJson ESP8266 легко витягує необхідні поля з відповіді API та використовує їх у виводі або аналізі.

ThingSpeak – одна з найпопулярніших IoT-платформ для зберігання і візуалізації даних. Вона дозволяє створювати персоналізовані канали з кількома полями, до яких ESP8266 надсилає дані через HTTP-запити. Візуалізація відбувається у вигляді графіків, які оновлюються автоматично. Крім того, можливе використання інструментів MATLAB для побудови аналітики та прогнозування трендів.

2.6 Висновок до розділу

У розділі було проведено технічне обґрунтування структури та функціональних можливостей кіберфізичної системи збору метеорологічних даних на базі мікроконтролера ESP8266. Розглянуто апаратну архітектуру проєкту, визначено склад основних сенсорних елементів, наведено їх метрологічні характеристики та способи інтеграції з обчислювальним модулем.

Окрему увагу приділено методам обробки та передачі даних у реальному часі. Забезпечено підтримку двох режимів роботи – портативного та стендового – міз можливістю передавання даних як локально (через USB), так і через бездротові канали (Wi-Fi). Реалізовано механізми виводу інформації на дисплей, а також

передачі результатів вимірювань на зовнішні сервіси, зокрема Narodmon, Dweet.io, Google Drive тощо. Це забезпечує високий рівень доступності й масштабованості розробленої системи.

Результати проєктування підтверджують відповідність системи вимогам до автономних метеостанцій, зокрема за параметрами енергоефективності, точності вимірювань, простоти обслуговування та інтеграції в інформаційно-аналітичні середовища. Отримані технічні рішення формують основу для реалізації наступного етапу – програмно-апаратної реалізації системи.

КІБЕРФІЗИЧНА СИСТЕМА ПРОГНОЗУВАННЯ ОПАДІВ НА ОСНОВІ ESP8266

3.1. Алгоритмізація та реалізація комплексу задач системи

Розробка функціональних схем є одним із ключових етапів створення сучасних інформаційних систем, зокрема метеорологічних. Вони визначають усі необхідні функції, які система повинна виконувати, і закладають основу для її ефективної реалізації.

Визначення функцію. Функціональні схеми дозволяють чітко окреслити набір функцій, які має реалізовувати система. Це формує загальне уявлення про її призначення, взаємодію між компонентами та логіку обробки даних.

Оптимізація роботи. Завдяки схемам розробники можуть на ранніх етапах виявити можливі слабкі місця у структурі або логіці роботи пристрою. Це дозволяє вчасно внести корективи, що сприяє стабільності, надійності та орієнтації системи на реальні потреби користувача.

Формування вимог. Схеми дають змогу деталізувати технічні та функціональні вимоги до системи: які параметри мають бути виміряні (температура, вологість, тиск, опади тощо), як вони повинні оброблятись, зберігатись та виводитись. Це особливо важливо для побудови системи на базі мікроконтролера (наприклад, ESP8266), де потрібно раціонально розподіляти ресурси.

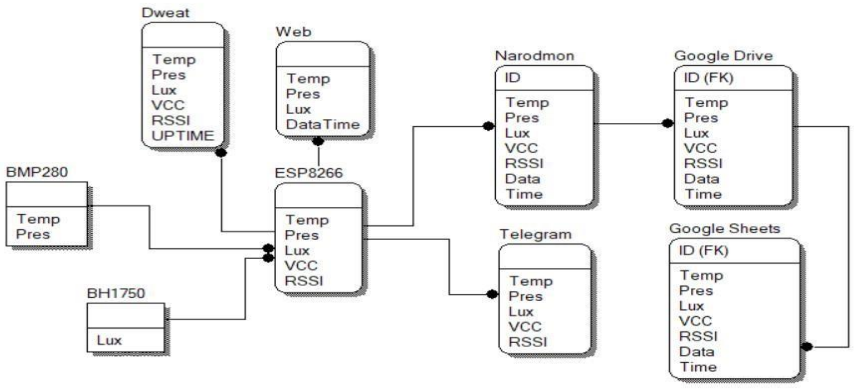


Рисунок 3.1 – Функціональна схема інформаційної системи

Функціональна схема розроблюваної інформаційної системи метеоспостереження охоплює три ключові рівні: отримання даних, їх обробку та формування вихідної інформації. На першому етапі система отримує вхідні дані у вигляді фізичних параметрів навколишнього середовища, які вимірюються за допомогою різних сенсорів, підключених до мікроконтролера. Це можуть бути значення температури, вологості, атмосферного тиску, освітленості та опадів. Усі ці показники надходять до мікроконтролера (наприклад, ESP8266), який обробляє сигнали з датчиків і переводить їх у цифрову форму, придатну для подальшого використання.

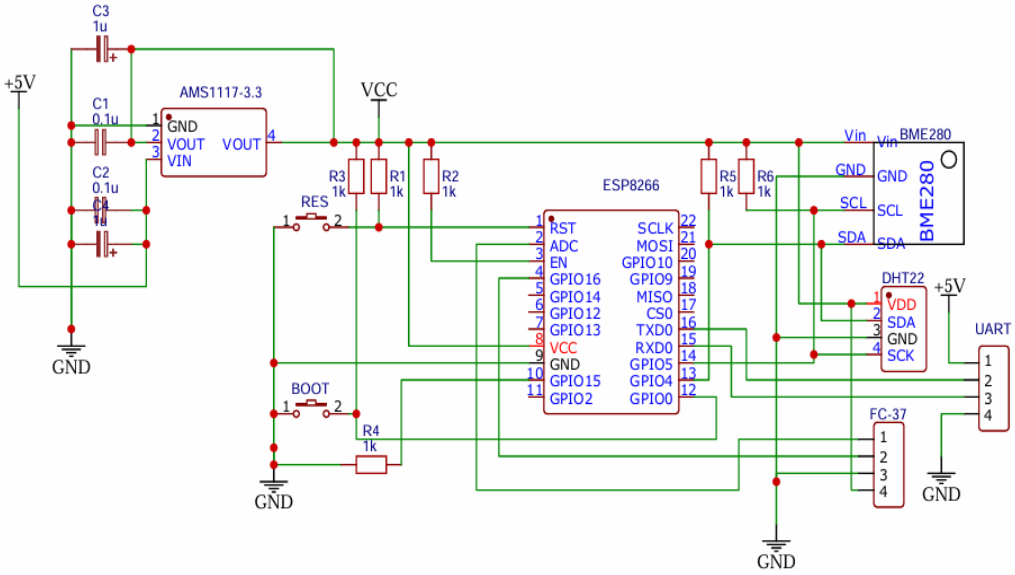


Рисунок 3.2 – Електрична принципова схема

На схемі зображено мікроконтролерну систему на базі модуля ESP8266, призначену для зчитування показників навколишнього середовища з цифрових сенсорів. Живлення здійснюється від джерела +5 В, напруга стабілізується до рівня 3,3 В за допомогою лінійного стабілізатора AMS1117-3.3. Для забезпечення електричної стабільності на вході й виході стабілізатора передбачено згладжувальні конденсатори C1–C3.

Мікроконтролер ESP8266 підключений до трьох периферійних пристроїв: цифрового сенсора температури та вологості DHT22, цифрового барометричного сенсора BME280 (через інтерфейс I²C), а також датчика вологості ґрунту FC-37. Для ініціалізації модуля ESP8266 передбачено кнопку скидання (RES) з підключеним резистором підтягування R3 і кнопку переведення в режим прошивки (BOOT) з резистором R4. Лінії GPIO0 і GPIO2 підключені відповідно до стандартних вимог до режимів роботи ESP8266.

Комунікація з сенсором BME280 здійснюється за протоколом I²C: лінії SCL і SDA підключено до GPIO5 і GPIO4 мікроконтролера через підтягувальні резистори R5 і R6 номіналом 1 кОм. Живлення модуля BME280 здійснюється від шини +3,3 В. DHT22 підключено до GPIO13, живиться від шини +5 В, передбачено підключення до спільної землі. FC-37 підключено до вільних GPIO ESP8266, відповідно до вибраної логіки керування.

Для налагодження системи передбачено UART-інтерфейс, що дає змогу здійснювати програмування та передачу даних через послідовний порт. Усі цифрові лінії підключено відповідно до вимог рівнів логіки ESP8266 (3,3 В).

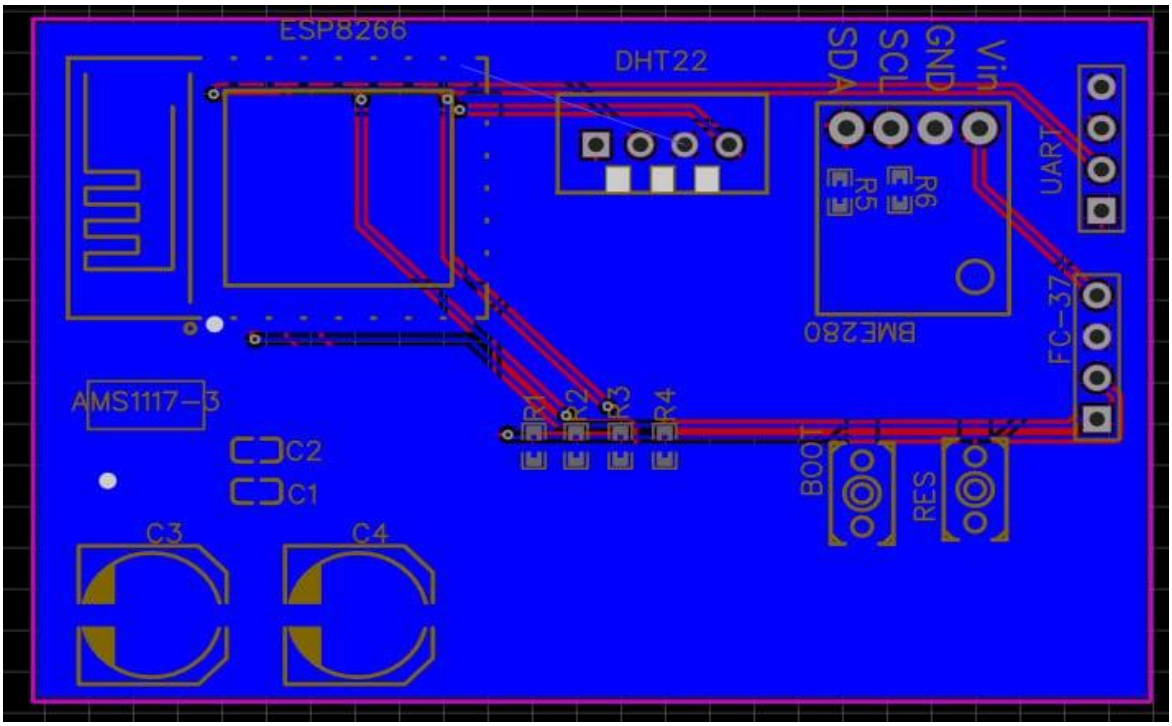


Рисунок 3.3 - Розроблена друкована плата

Розроблена друкована плата є односторонньою та призначена для реалізації мікроконтролерної системи на базі ESP8266 із сенсорами навколишнього середовища. Розміщення компонентів виконано з урахуванням мінімізації довжини критичних сигнальних трас і забезпечення стабільної роботи цифрових інтерфейсів.

Стабілізація напруги здійснюється за допомогою лінійного стабілізатора AMS1117-3.3, який живить мікроконтролер і сумісні периферійні пристрої. На вході й виході стабілізатора встановлено байпасні конденсатори (C1–C3), що забезпечують фільтрацію високочастотних шумів. Живлення подається окремими доріжками до кожного вузла для уникнення падінь напруги.

Мікроконтролер ESP8266 підключено до цифрового сенсора температури й вологості DHT22, барометричного сенсора BME280 (через інтерфейс I²C), а також до датчика вологості ґрунту FC-37. Для забезпечення коректної роботи шини I²C передбачені підтягувальні резистори R5 і R6. Комунікаційні лінії між ESP8266 та периферією прокладено прямолінійно з дотриманням мінімальної довжини та уникненням гострих кутів.

Кнопки керування BOOT та RES підключено до відповідних GPIO мікроконтролера через резистори підтягування. Вони винесені в окрему область плати для забезпечення зручного доступу при прошивці та налагодженні. Інтерфейс UART реалізовано у вигляді чотириконтактного роз’єму з доступом до TX, RX, живлення та землі, що дозволяє з’єднання з адаптером USB-UART.

Конструктивне рішення передбачає формування суцільного GND-полігона з метою зниження електричного шуму та забезпечення стабільного заземлення, що відповідає вимогам стандартів трасування цифрових високочастотних пристроїв.

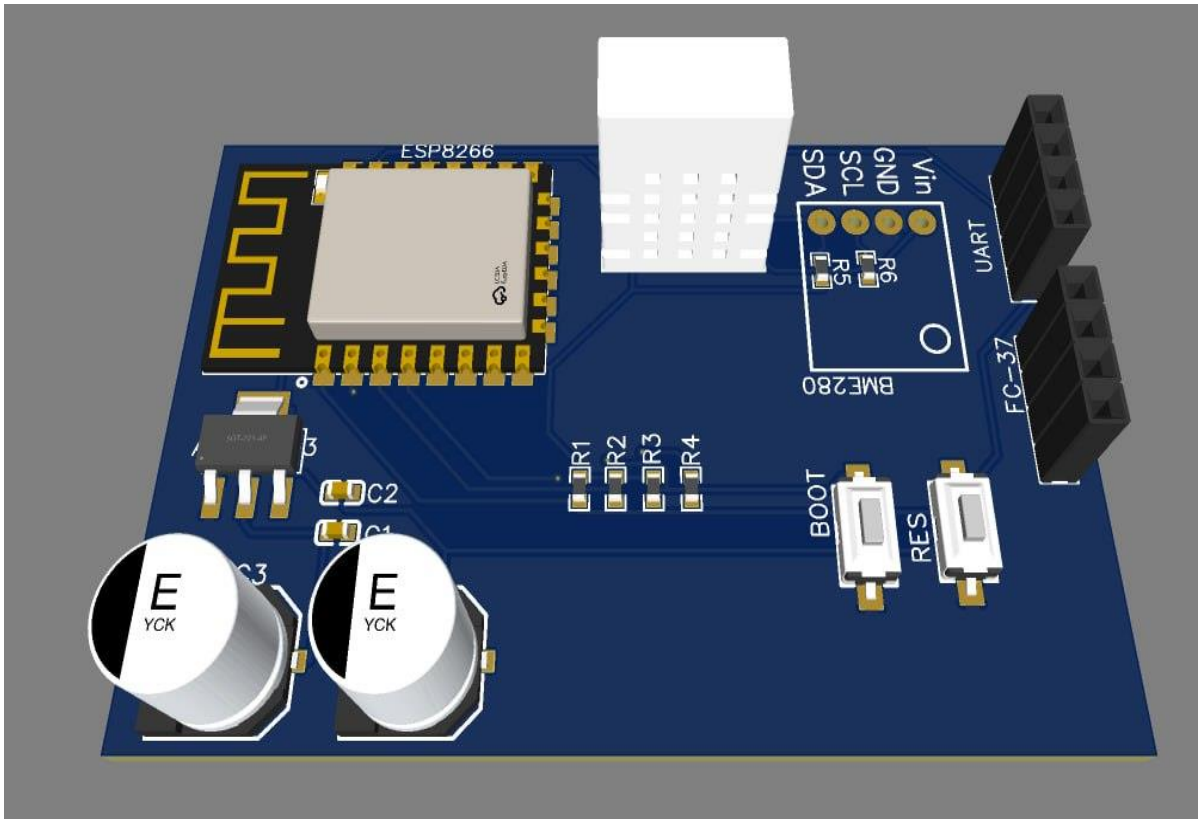


Рисунок 3.4 – 3D вигляд плати

Тривимірна модель плати відображає компонування та фізичну реалізацію мікроконтролерного модуля з інтеграцією основних елементів схеми. Модуль ESP8266 розташований центрально, що дозволяє симетричне трасування ліній GPIO до периферійних пристроїв. Контактні майданчики модуля мають позолочене покриття типу ENIG (Electroless Nickel Immersion Gold), що підвищує надійність з'єднань і захист від корозії.

Поблизу стабілізатора AMS1117-3.3 змонтовані конденсатори C2, C3 і C4, які забезпечують фільтрацію імпульсних завад і стабілізацію вихідної напруги. Паралельне розміщення електролітичних і керамічних конденсаторів дозволяє ефективно придушувати як низько-, так і високочастотні компоненти шумів. Стабілізатор розміщено з урахуванням тепловідведення на полігон землі.

Сенсори DHT22 і BME280 змонтовані з дотриманням механічної орієнтації, що забезпечує зручність монтажу та заміни. Інтерфейс I²C для BME280 реалізовано через лінії SDA і SCL, виведені до контактної групи з відповідними отворами під

пайку. Додані підтягувальні резистори R5 і R6 встановлені поблизу сенсора для зменшення паразитних імпедансів.

Кнопки BOOT і RES виконані у вигляді тактильних елементів з поверхневим монтажем (SMD), розміщені на краю плати для зручного доступу. Контактні роз'єми UART і FC-37 винесені вертикально для можливості підключення зовнішніх модулів через стандартні штифтові роз'єми (2.54 мм pitch).

Усі елементи змонтовано на верхньому боці друкованої плати. Сигнальні доріжки прокладено з дотриманням вимог до мінімального радіусу згину, мінімальної ширини провідників для струмових ліній, а також з урахуванням правил розміщення за стандартом IPC-2221. Топологія відповідає вимогам до низьковольтних пристроїв з живленням 3,3–5 В.

На етапі обробки інформації система здійснює кілька паралельних процесів. Вона надсилає зібрані метеорологічні дані на загальнодоступні сервіси, як-от IoT-платформи для візуалізації чи зберігання, та синхронізує їх із веб-сервером, через який здійснюється доступ до інформації в локальній або глобальній мережі. Крім того, система обробляє запити користувачів, отримані через Telegram-бота: зберігає історію запитів і формує відповіді на основі актуальних вимірів.

Завершальним етапом функціональної схеми є формування вихідних даних. Результати вимірів виводяться на веб-сервер у зручному графічному або текстовому форматі, доступному для перегляду з будь-якого пристрою. Користувач також отримує інформацію у відповідь на свої запити в Telegram. Паралельно система автоматично зберігає отримані дані у хмарному сховищі Google Drive з можливістю подальшого опрацювання та аналізу у Google Sheets. Такий підхід забезпечує не лише зручність у доступі до даних, але й їх збереження для довгострокового використання.

Для розробки програмного забезпечення було обрано середовище Arduino IDE. Це програмне середовище спеціалізується на створенні застосунків для мікроконтролерів і вирізняється зручним та інтуїтивно зрозумілим інтерфейсом. Воно забезпечує всі необхідні інструменти для написання, компіляції,

завантаження коду та взаємодії з апаратною частиною. Arduino IDE підтримує базові функції редагування, компіляції та налагодження програмного коду. Завдяки великій спільноті користувачів, платформа має широкі ресурси для навчання й обміну досвідом. Ця розробницька платформа є однаково зручною як для новачків, так і для досвідчених програмістів.

Початковим етапом стало підключення всіх необхідних бібліотек для реалізації системи. Деякі з них вже інтегровані в середовище розробки Arduino IDE, тоді як інші довелося встановити вручну за допомогою вбудованих інструментів самої платформи.

Інтерфейс вбудованого менеджера бібліотек у середовищі розробки Arduino IDE. Інтерфейс містить панель фільтрації, яка дозволяє здійснювати пошук за ключовими словами, а також відфільтровувати бібліотеки за типом і тематикою. У наведеному прикладі обидва параметри фільтрації встановлені на значення "Всі", що, ймовірно, позначає вузьку категорію бібліотек, пов’язаних із певною тематикою чи проєктом.

У результаті відображено бібліотеку AIPlc_Opta, яка призначена для використання з контролером Arduino Opta. Бібліотека реалізує PLC-функціональність (програмовану логіку керування) в середовищі Arduino IDE. Це runtime-бібліотека, яка включає плагіни для забезпечення підтримки функцій, необхідних для виконання програм логічного керування. Існує можливість обрати доступну версію бібліотеки (у даному випадку – 1.1.0) та ініціювати її встановлення через інтерфейс користувача IDE. Зазначена функціональність дозволяє інтегрувати необхідні бібліотеки безпосередньо із середовища розробки, що спрощує процес конфігурації програмного забезпечення для мікроконтролерів Arduino.

До початку основної логіки програми здійснюється імпорт бібліотек, необхідних для функціонування всіх апаратних і мережевих компонентів системи, реалізованої на мікроконтролері ESP8266.

Бібліотека ESP8266WiFi.h забезпечує доступ до функцій налаштування бездротового інтерфейсу та управління з'єднанням у режимі Wi-Fi Station або Access Point відповідно до стандарту IEEE 802.11 b/g/n. Для реалізації TCP-з'єднань використовується модуль WiFiClient.h, який дозволяє організувати передачу даних між ESP8266 та зовнішніми вузлами. У разі необхідності ініціалізації SSL/TLS-з'єднань за протоколом HTTPS підключається бібліотека WiFiClientSecure.h, яка реалізує підтримку X.509-сертифікатів та перевірку автентичності серверів, зокрема для Telegram Bot API.

Компонент ESP8266WebServer.h дозволяє створити вбудований HTTP-сервер із підтримкою обробки запитів методів GET і POST. Для реалізації клієнтської логіки HTTP-запитів підключається ESP8266HTTPClient.h. Бібліотека ESP8266mDNS.h дає змогу використовувати Multicast DNS (mDNS), що дозволяє доступ до пристрою через доменне ім'я локальної мережі, наприклад, esp.local, замість статичної IP-адреси.

Для реалізації інтерфейсу з периферійними пристроями, які працюють через послідовну шину I²C (згідно зі стандартом NXP I2C-bus specification), використовується Wire.h. Сенсорна бібліотека Adafruit_Sensor.h забезпечує уніфікований інтерфейс для доступу до фізичних значень з усіх сумісних пристроїв виробництва Adafruit. Для взаємодії з цифровим барометричним сенсором BMP280 використовується Adafruit_BMP280.h, а з фотометричним сенсором освітленості BH1750 – BH1750.h.

Підключення Telegram-бота реалізується через бібліотеку UniversalTelegramBot.h, яка дозволяє виконувати автентифікацію за токеном, надсилати повідомлення, обробляти команди та взаємодіяти з Telegram Bot API через HTTPS. Для обробки структурованих даних у форматі JSON застосовується ArduinoJson.h, що забезпечує парсинг, генерацію та модифікацію об'єктів типу JsonDocument.

Оголошення бібліотек на початку програми є обов'язковою умовою для подальшої компіляції коду, коректного створення об'єктів класів і виклику

відповідних методів, які безпосередньо взаємодіють з апаратними або мережевими ресурсами системи.

ADC мікроконтролера ESP8266 конфігуровано в режимі ADC_VCC, що дозволяє вимірювати напругу живлення без використання зовнішніх аналогових входів. Для збору сенсорних даних використовуються цифрові датчики BMP280 і BH1750, з'єднані через шину I2C. Датчик BMP280 забезпечує отримання значень атмосферного тиску та температури, а BH1750 – рівня освітленості в люксах.

Мережева взаємодія реалізована через об'єкт WiFiClient, який використовується для TCP-з'єднань, а також через локальний HTTP-сервер на порті 80 на базі бібліотеки ESP8266WebServer. Для захищених запитів до зовнішніх API застосовується об'єкт WiFiClientSecure, що підтримує TLS і верифікацію сертифікатів за допомогою об'єкта X509List.

Передбачено два канали передачі телеметричних даних: перший – на сервер Narodmon (адреса eu.narodmon.com, порт 8283, TCP), другий – на платформу dweet.io (порт 80, HTTP). Трансмісія даних на Narodmon здійснюється з періодичністю 300 000 мс, що відповідає 5 хвилинам, а проміжок між регулярними вимірюваннями встановлено на 5000 мс.

Об'єкт UniversalTelegramBot ініціалізовано з токеном доступу до Telegram Bot API, що забезпечує асинхронну взаємодію з користувачем через захищене з'єднання. Для встановлення TLS-з'єднання використовується кореневий сертифікат Telegram.

Оголошені змінні призначені для збереження поточних значень: vcc – напруга живлення, T – температура, P – тиск, L – освітленість, rssi – рівень сигналу Wi-Fi. Змінні типу unsigned long застосовуються для реалізації таймерів, які керують інтервалами збору та надсилання даних.

Функція ADC_MODE(ADC_VCC) конфігурує аналогово-цифровий перетворювач мікроконтролера ESP8266 на вимірювання напруги живлення самого мікроконтролера, що дозволяє проводити моніторинг рівня Vcc без зовнішніх з'єднань. Об'єкт Adafruit_BMP280 bmp відповідає за ініціалізацію сенсора BMP280,

призначеного для вимірювання атмосферного тиску та температури. Комунікація з цим сенсором здійснюється за допомогою шини I2C. Шина I2C (Inter-Integrated Circuit) – це дволінійна синхронна послідовна шина зв’язку, яка використовується для з’єднання мікроконтролерів із периферійними пристроями. Вона складається з двох сигнальних ліній: SDA (Serial Data Line) – для передавання даних, і SCL (Serial Clock Line) – для синхронізації. Обидві лінії є відкритими колекторами й потребують підключення підтягувальних резисторів до джерела живлення.

Об’єкт BH1750 lightMeter інкапсулює функціональність цифрового сенсора освітленості BH1750, який також працює через інтерфейс I2C. Для реалізації TCP-клієнта, який використовується під час передавання даних або при з'єднанні з віддаленими сервісами, створюється екземпляр WiFiClient client. Компонент ESP8266WebServer server(80) ініціалізує вебсервер, що працює на порту 80 і забезпечує базову HTTP-взаємодію.

Константи host, httpPort, dweet та dweetPort задають параметри віддалених серверів обміну даними: Narodmon (eu.narodmon.com, порт 8283) та Dweet.io (порт 80). Значення interval = 5000 визначає періодичність виконання основних операцій – 5000 мс. Константа BOT_TOKEN містить токен авторизації Telegram-бота, необхідний для ініціалізації об'єкта UniversalTelegramBot.

Компонент X509List cert(TELEGRAM_CERTIFICATE_ROOT) задає набір кореневих сертифікатів для верифікації TLS-з'єднань. Для забезпечення захищеної передачі даних створюється об'єкт WiFiClientSecure secured_client, який використовується при роботі з Telegram API.

Об'єкт bot класу UniversalTelegramBot інкапсулює функціональність взаємодії з Telegram-ботом через TLS. Змінні типу float – vcc, T, P, L – призначені для збереження поточних значень напруги, температури, тиску та освітленості відповідно. Змінна rssi типу int зберігає поточний рівень сигналу Wi-Fi. Таймери типу unsigned long, такі як lasttime та lastNarodmonSendTime, використовуються для керування інтервалами між подіями. Константа narodmonSendInterval = 5 * 60 *

1000 встановлює інтервал надсилання даних до сервера Narodmon тривалістю 5 хвилин.

Щоб дані на веб-сторінці відображалися правильно, необхідно сформувати коректні запити та забезпечити привабливе оформлення самої сторінки.

HTML-документ завершується вставленням закривальних тегів </body> та </html>, після чого функція повертає згенерований рядок, що містить повну структуру вебсторінки. Для обробки запитів до API реалізовано функцію handleData(), яка формує відповідь у форматі JSON. Для цього створюється об’єкт StaticJsonDocument з обсягом пам’яті 200 байт, у якому записуються поточні значення температури (T), атмосферного тиску (P) та освітленості (L). Отримана структура серіалізується у рядок jsonData за допомогою функції serializeJson(), після чого надсилається клієнту з HTTP-статусом 200 і заголовком Content-Type: application/json. Обробку запитів до HTML-інтерфейсу виконує функція handleWebpage(), яка ініціює відповідь з HTTP-статусом 200 та типом вмісту text/html, використовуючи результат роботи функції generateWebpage() як тіло відповіді.Функція generateWebpage() створює HTML-сторінку з використанням CSS для стилізації та JavaScript для інтерактивності. Сторінка включає блоки для виведення метеорологічних показників та поточної дати й часу.

3.2 Підключення програмного забезпечення до мережі

Підключення до WiFi-мережі є одним із критично важливих процесів у функціонуванні метеостанції на базі мікроконтролера ESP8266. Стабільна передача даних на вебсервери, хмарні сервіси чи до Telegram-бота можлива лише за умови надійного інтернет-з’єднання. Для досягнення цього в проєкті було реалізовано кілька функціональних рішень, які охоплюють різні сценарії підключення.

Перш за все, реалізовано можливість підключення до заданої мережі, дані про яку (назва та пароль) зберігаються у програмному коді. Це забезпечує швидкий

старт пристрою в передбачуваному середовищі, наприклад, у домашній або лабораторній мережі.

Окрім того, реалізовано алгоритм сканування доступних WiFi-мереж. У випадках, коли підключення до основної мережі неможливе (наприклад, через зміну конфігурації або відсутність сигналу), мікроконтролер здійснює пошук усіх наявних точок доступу. За певними критеріями (сила сигналу, наявність збережених облікових даних) він може автоматично спробувати встановити з’єднання з однією з них.

Для забезпечення надійності роботи системи додатково реалізовано функцію перевірки статусу з’єднання. Вона періодично оцінює, чи активне з’єднання з мережею, і у випадку його втрати – ініціює повторне підключення. Такий підхід дозволяє уникнути тривалих простоїв і забезпечити безперервну передачу метеоданих.

Таким чином, підключення до WiFi реалізовано з урахуванням різних умов експлуатації пристрою, що значно підвищує його стабільність і функціональність у реальних сценаріях використання.

Було реалізовано функцію, яка дозволяє автоматично підключатися до заздалегідь налаштованої мережі. Її створено для забезпечення доступу метеостанції до Інтернету у випадках, коли користувач не має можливості безпосередньо під'єднатися до пристрою для введення параметрів своєї мережі.

Функція connectToPreferredWiFi() реалізує підключення мікроконтролера ESP8266 до заздалегідь визначеної Wi-Fi мережі з відомими параметрами аутентифікації. Ідентифікатор мережі (SSID) та пароль задаються у вигляді символьних констант типу const char* і передаються до функції WiFi.begin(), яка ініціалізує процес встановлення з'єднання відповідно до стандарту IEEE 802.11 b/g/n.

Контроль статусу з’єднання здійснюється в циклі перевірки повернення значення WL_CONNECTED методом WiFi.status(). Для обмеження часу очікування застосовується механізм відліку з використанням функції millis(). Якщо протягом

10 000 мс підключення не встановлено, виконується примусове завершення функції без подальшого виконання мережевих операцій. Такий підхід мінімізує затримки у випадку недоступності точки доступу.

У разі успішного з’єднання через серіальний порт виводиться IP-адреса, отримана динамічно через DHCP-сервер маршрутизатора. Для цього використовується метод WiFi.localIP().toString(), який формує представлення адреси у форматі IPv4 у вигляді текстового рядка. Цей механізм дозволяє забезпечити ідентифікацію вузла в локальній мережі та надалі використовувати IP-адресу для обробки HTTP-запитів або передачі телеметрії.

У функції вказані параметри мережі, до якої необхідно здійснити підключення. Якщо підключення проходить успішно, в консоль виводяться відповідні повідомлення. У разі, якщо з’єднання не встановлюється протягом 10 секунд, виконання функції припиняється, і система переходить до наступного етапу підключення через іншу функцію.

const char* preferredSSID = "vallyk97"; та const char* preferredPassword = "48973592" - У коді зберігаються назва мережі та пароль. Це означає, що після запуску пристрій автоматично буде намагатися з'єднатися саме з цією мережею – без додаткового втручання користувача.

Виклик WiFi.begin(preferredSSID, preferredPassword) ініціалізує процедуру встановлення з’єднання мікроконтролера ESP8266 з мережею Wi-Fi за вказаними параметрами SSID та паролем. Після запуску функції відбувається негайна спроба встановити з'єднання з точкою доступу відповідно до протоколу IEEE 802.11.

Контроль стану з'єднання реалізований за допомогою циклу з умовою WiFi.status() != WL_CONNECTED, яка перевіряє, чи не змінено статус на WL_CONNECTED, що свідчить про успішне з’єднання. Під час кожної ітерації здійснюється затримка тривалістю 500 мс для зниження частоти опитування та уникнення перевантаження Wi-Fi стека.

Вимірювання тривалості спроби підключення реалізується за допомогою функції millis(), яка обчислює кількість мілісекунд від моменту запуску пристрою.

Якщо з моменту початку з’єднання пройшло понад 10 000 мс, з’єднання вважається невдалим, про що інформується через інтерфейс Serial, після чого функція завершує виконання без продовження роботи з мережею. Такий підхід дозволяє обмежити час очікування з’єднання та забезпечити керованість поведінки пристрою в разі відсутності доступу до мережі.

Serial.println("Успішно підключено..."); та Serial.println("Ваш IP-адреса " + WiFi.localIP().toString()) - Якщо все пройшло успішно, система повідомляє про це та виводить локальну IP-адресу. Це дає змогу користувачу бачити, що пристрій підключений і готовий до роботи (наприклад, доступ до веб-сервера).

Якщо не відбулося підключення до попердньо заданої мережі, система переходить до наступної функції підключення до мережі.

Функція connectWiFi() реалізує алгоритм автоматизованого підключення пристрою до бездротової мережі з fallback-логікою у разі недоступності основної конфігурованої точки доступу. На початковому етапі викликається функція connectToPreferredWiFi(), яка ініціює підключення до попередньо визначеної мережі за жорстко закодованими обліковими даними (SSID та пароль). Якщо через WiFi.status() виявлено, що з’єднання не встановлено (WL_CONNECTED), виконується перехід до процедури сканування навколишніх мереж.

Для цього встановлюється режим роботи WIFI_STA, здійснюється розрив поточного з’єднання через WiFi.disconnect(), після чого викликається WiFi.scanNetworks() для виявлення доступних SSID у зоні покриття. Результат сканування виводиться через інтерфейс Serial, зокрема, список доступних мереж за допомогою WiFi.SSID(i).

Після цього очікується введення параметрів цільової мережі (SSID та пароль) з консолі користувача через Serial.readStringUntil('\n'). Отриманий SSID перевіряється на наявність у переліку знайдених мереж. Якщо відповідність відсутня, цикл сканування повторюється.

У разі виявлення SSID виконується спроба підключення за допомогою WiFi.begin(), з подальшим контролем статусу підключення у циклі з граничним

часом очікування 10 секунд. Якщо протягом цього часу підключення не підтверджено, виконується повторна спроба. При успішному з’єднанні функція виводить IP-адресу пристрою, отриману через DHCP (виклик WiFi.localIP().toString()), після чого виходить з циклу.

Таким чином, реалізовано механізм підключення з пріоритетним доступом до визначеної мережі та резервним сценарієм ручного вибору мережі за результатами активного сканування. Це забезпечує стабільність функціонування бездротового інтерфейсу в умовах змінної доступності точок доступу.

У цій функції реалізовано процес сканування доступних мереж, після чого у консоль виводиться перелік знайдених WiFi-мереж. Користувач повинен ввести назву бажаної мережі та відповідний пароль. Якщо підключення не вдається, система повідомляє про помилку в консолі та повторює спробу з'єднання у новому циклі.

Функція connectToPreferredWiFi() викликається з метою встановлення з’єднання з наперед визначеною мережею Wi-Fi, для якої заздалегідь задано значення SSID і пароля. Якщо з’єднання встановлюється успішно (статус WL_CONNECTED), подальше виконання процедури підключення припиняється. У випадку відсутності з’єднання ініціюється процедура сканування оточення для виявлення доступних бездротових мереж.

Режим модуля встановлюється в WIFI_STA, що відповідає клієнтському режиму відповідно до стандарту IEEE 802.11. Попереднє з’єднання скидається викликом WiFi.disconnect(). Для отримання списку доступних мереж виконується функція WiFi.scanNetworks(). Усі знайдені мережі виводяться через UART з використанням функції Serial.println(WiFi.SSID(i)).

Користувач вводить ідентифікатор SSID та відповідний пароль через інтерфейс Serial, що дає змогу встановити з’єднання з будь-якою мережею без потреби модифікації прошивки. Введені параметри зчитуються через Serial.readStringUntil('\n'). Далі перевіряється наявність введеного SSID серед

знайдених точок доступу. У разі відсутності відповідності процедура сканування повторюється.

Якщо SSID знайдено, виконується спроба підключення за допомогою WiFi.begin(). Контроль успішності підключення реалізується в циклі з тайм-аутом 10 секунд, що дозволяє уникнути блокування виконання програми в разі відсутності відповіді з боку точки доступу. За умови встановлення з'єднання система отримує IP-адресу через DHCP, яка виводиться через WiFi.localIP().toString().

Функція checkWiFiConnection() виконує моніторинг статусу з'єднання. Якщо WiFi.status() не дорівнює WL_CONNECTED, відбувається виклик connectWiFi() для повторного встановлення зв'язку. Цей механізм забезпечує автоматичне відновлення підключення в разі його втрати, що критично для безперервної роботи системи, орієнтованої на передавання телеметричних даних або взаємодію з мережевими службами.

Підключення до Telegram Bot є одним із способів передавання метеоданих. У заздалегідь створеному боті були налаштовані команди для отримання інформації з системи. Цей процес можна умовно поділити на два основні етапи:

– встановлення з'єднання з Telegram API;

– обробка вхідної інформації, отриманої від бота.

Telegram – це один із найпоширеніших месенджерів, який надає можливість обміну повідомленнями, а також передавання аудіо- та відеофайлів. Основною його перевагою є система шифрування, що гарантує безпеку та конфіденційність даних користувачів. Завдяки відкритому API, розробники мають змогу інтегрувати свої сервіси та додатки з платформою.

Функція setupTelegram() виконує ініціалізацію параметрів, необхідних для встановлення захищеного TLS-з'єднання з Telegram Bot API через інтерфейс WiFiClientSecure. На першому етапі викликається метод setTrustAnchors(&cert), який додає кореневий сертифікат для домену api.telegram.org у контекст TLS-клієнта. Це забезпечує верифікацію ланцюга довіри відповідно до стандарту X.509.

Далі ініціалізується синхронізація системного часу за допомогою виклику configTime(0, 0, "pool.ntp.org"), що використовує зовнішній NTP-сервер. Отримання достовірного часу є критично необхідним для коректної обробки HTTPS-запитів, оскільки TLS-сертифікати перевіряються з урахуванням їх строку дії. Контроль завершення синхронізації реалізовано через змінну time_t now, яка періодично оновлюється у циклі до досягнення значення, більшого за 86 400 секунд (одна доба в секундах), що підтверджує коректність отриманого часового штампу.

Після завершення процедури система готова до автентифікованої взаємодії з Telegram API через захищене з’єднання.

Завдяки використанню заздалегідь визначених констант і попередньо ініціалізованих об’єктів, реалізована функція підключення до Telegram-бота є не лише компактною, а й зрозумілою для подальшого супроводу. Вона містить повідомлення, які виводяться у консоль та інформують про початок з’єднання, хід обробки та успішне завершення комунікації.

Підключення до Telegram API відбувається через захищений канал (HTTPS), що істотно знижує ризики перехоплення даних чи несанкціонованого доступу. Це критично важливо в умовах передавання чутливої інформації, навіть якщо йдеться лише про кліматичні параметри.

Заздалегідь був створений Telegram Bot, що містить основні команди для отримання метеорологічних даних. Після його створення було отримано BOT_TOKEN, який використовується для зв’язку з ботом. Цей токен, необхідний для коректної роботи, було оголошено на початку програми.

У Telegram-боті реалізовано клавіатурне меню, яке забезпечує доступ до функцій зчитування сенсорних даних через текстові команди. Команда /temp використовується для запиту температурного значення, яке зчитується з датчика BMP280. Команда /pres ініціює отримання значення атмосферного тиску з того ж сенсора. Для визначення рівня освітленості передбачена команда /lux, що викликає звернення до фотодатчика BH1750. Команда /all повертає зведену інформацію з усіх підключених сенсорів, включно з температурою, тиском, освітленістю, а

також, за потреби, рівнем сигналу Wi-Fi (RSSI) і напругою живлення (Vcc). Команда /help надає коротку текстову інструкцію щодо доступних команд і формату взаємодії з ботом.

Обробка запитів реалізована в основному циклі програми через порівняння вхідного тексту з наперед визначеними командами. У разі збігу формується відповідне текстове повідомлення з MIME-типом text/plain, яке передається до Telegram-сервера через захищене TLS-з'єднання. Застосування Telegram Bot API забезпечує інтерактивний доступ до даних, що зчитуються в реальному часі з сенсорного модуля мікроконтролерної системи.

Функція handleNewMessages(int numNewMessages) реалізує обробку масиву вхідних повідомлень, отриманих через Telegram Bot API, з використанням об'єкта UniversalTelegramBot. Вхідний параметр numNewMessages вказує кількість повідомлень, доступних для обробки в поточному сеансі. Кожне повідомлення обробляється в циклі, де на основі значення bot.messages[i].text виконується вибір відповідної дії.

У разі, якщо текст повідомлення дорівнює /temp, генерується відповідь із зазначенням поточної температури (T) у градусах Цельсія. Команда /pres повертає атмосферний тиск (P) у мм рт. ст., а /lux – рівень освітленості (L) у люксах. Для кожного з цих запитів створюється відповідний текстовий рядок і передається методом sendMessage() до чату, ідентифікованого через chat_id, що унеможливлює пересилання відповідей стороннім користувачам.

У випадку команди /all формується повідомлення з агрегованими даними: температурою, тиском, освітленістю, напругою живлення (vcc) та рівнем сигналу Wi-Fi, зчитаним функцією WiFi.RSSI(). Повідомлення передається у форматі Markdown.

Команда /help викликає текстову відповідь, яка містить перелік підтримуваних команд з коротким описом їх функціонального призначення. Обробка повідомлень, що не відповідають жодній з визначених команд, не виконується.

Комунікація з Telegram API здійснюється через захищене TLS-з'єднання з використанням об'єкта WiFiClientSecure, що забезпечує автентичність, цілісність та конфіденційність переданих даних.

3.3 Вимірювання метрологічних параметрів

Метрологічні дані є ключовим елементом функціонування даної системи, тому стабільна й точна робота сенсорів має вирішальне значення. У цьому проєкті використовуються два датчики для збору метеоданих:

- BMP280
- BH1750

BMP280 – високоточний сенсор, призначений для вимірювання температури та атмосферного тиску в навколишньому середовищі [5]. Він може фіксувати температуру в межах від -40°C до +85°C, а тиск – у діапазоні від 300 гПа до 1100 гПа. Підключення можливе через інтерфейси I2C або SPI.

BH1750 – цифровий сенсор, який вимірює рівень освітленості [6]. Завдяки високій точності, він ефективно працює в широкому спектрі умов освітлення. Як і BMP280, підтримує підключення через шини I2C або SPI.

У нашому випадку вибрано підключення через I2C, оскільки цей варіант є зручнішим і простішим у використанні.

Вимірювання температури та атмосферного тиску здійснюється за допомогою датчика BMP280, тому його коректна ініціалізація є обов’язковою умовою для стабільної роботи всієї системи.

Функція measure() виконує однократне зчитування значень з сенсорних модулів та внутрішніх джерел даних мікроконтролера ESP8266. Ініціалізація шини I²C здійснюється за допомогою виклику Wire.begin(), що забезпечує доступ до периферійних пристроїв, підключених через відповідний інтерфейс.

Вимірювання напруги живлення мікроконтролера виконується методом ESP.getVcc(), який повертає результат у мілівольтах. Для перетворення у вольти

значення типу uint16_t приводиться до float і масштабується поділом на 1000. Отримане значення записується у змінну vcc.

Рівень сигналу бездротового інтерфейсу Wi-Fi визначається методом WiFi.RSSI(), який повертає поточне значення RSSI (Received Signal Strength Indicator) у децибелах відносно мілівата (dBm). Результат зберігається у змінній rssi.

Читання температури та атмосферного тиску здійснюється з використанням цифрового сенсора BMP280. Метод bmp.readTemperature() повертає значення температури у градусах Цельсія та присвоюється змінній T. Метод bmp.readPressure() повертає значення тиску в паскалях. Для приведення до міліметрів ртутного стовпчика використовується коефіцієнт перерахунку: P = P / 133.322, результат зберігається у змінній P.

Отримані дані передаються на UART через Serial.print() у структурованому форматі: температура (°C), атмосферний тиск (мм рт. ст.), напруга живлення (V), рівень сигналу (dBm). Такий формат виводу забезпечує зручність під час діагностики, а також може бути використаний для логування або тестування системи у процесі розробки.

Яскравість освітлення є третім вимірюваним параметром у структурі метеосистеми, поряд із температурою та атмосферним тиском. Для вимірювання освітленості використовується цифровий фотометр BH1750, що підключений до мікроконтролера через інтерфейс I²C. Ініціалізація шини здійснюється викликом Wire.begin(), аналогічно до ініціалізації сенсора BMP280.

Після встановлення з’єднання з пристроєм виконується ініціалізація BH1750 командою lightMeter.begin(), яка переводить датчик у режим безперервного вимірювання освітленості з автоматичним налаштуванням часу інтеграції. Зчитування значення освітленості реалізується через метод lightMeter.readLightLevel(), результат якого зберігається у змінній L типу float. Значення повертається у люксах відповідно до специфікації пристрою (I²C цифровий вихід, 1 lx роздільна здатність).

Зчитане значення передається у UART через виклики Serial.print() у форматі L= <значення> Lx, що забезпечує базову візуалізацію результату в режимі реального часу. Отримані дані можуть бути використані для локального аналізу, автоматизованого контролю стану середовища або відправлення до зовнішніх систем через бездротові канали зв'язку.

3.4 Поширення отриманих даних в загальний доступ

Публічне оприлюднення зібраних метеорологічних даних є доцільним для реалізації механізмів візуалізації та подальшої інтеграції системи у відкриті інформаційні середовища. З цією метою доцільно використовувати спеціалізовані онлайн-платформи, призначені для публікації телеметричних даних у реальному часі. Такі сервіси, як Narodmon та Dweet.io підтримують передачу даних через стандартні HTTP-запити та забезпечують доступ до API для інтеграції з іншими системами. Застосування подібних платформ спрощує обмін даними між користувачами, дозволяє формувати репрезентативні вибірки для аналітики, а також сприяє масштабуванню розроблених рішень за рахунок участі у глобальних спільнотах розробників.

Платформа Narodmon призначена для агрегування, обміну та моніторингу даних про параметри навколишнього середовища, зокрема метеорологічні та екологічні показники. Інфраструктура сервісу передбачає можливість підключення користувацьких сенсорних пристроїв через стандартні мережеві протоколи, зокрема HTTP/HTTPS, з подальшою публікацією отриманих даних у відкритому або обмеженому доступі. Платформа підтримує створення індивідуальних проєктів із прив'язкою до облікових записів користувачів, а також дозволяє реалізовувати сценарії обміну даними між різними пристроями спільноти.

Ключовим функціональним елементом є інтерактивна карта з візуалізацією показників у реальному часі, що забезпечує базову просторову аналітику. Такий підхід дозволяє проводити регіональний аналіз розподілу температури, вологості,

атмосферного тиску, рівня забруднення тощо, що є релевантним для задач моніторингу, дослідження мікрокліматичних умов або валідації локальних моделей погоди. Завдяки використанню точних сенсорів (з типовою похибкою у межах ±0,1...±0,5 одиниць виміру), дані, отримані з платформи Narodmon, можуть бути використані як джерело емпіричних спостережень у наукових дослідженнях.

З огляду на ці можливості, одним із функціональних завдань розробленої системи було забезпечення автоматизованого передавання зібраних метеоданих на платформу Narodmon у форматі, що відповідає її технічним вимогам.

Функція narodmonSend() реалізує передачу телеметричних даних на сервер системи Narodmon з використанням TCP-з’єднання. Надсилання здійснюється лише в разі перевищення встановленого інтервалу narodmonSendInterval, контроль якого реалізовано через порівняння поточного значення millis() зі змінною lastNarodmonSendTime.

У разі досягнення необхідного інтервалу здійснюється ініціалізація з’єднання з віддаленим хостом host на порту httpPort методом client.connect(). Якщо з'єднання не може бути встановлено, функція завершується з відповідним повідомленням у послідовний порт.

У разі успішного з'єднання формується пакет згідно з протоколом Narodmon. Першим передається ідентифікатор у вигляді MAC-адреси пристрою (WiFi.macAddress()), далі – мітка пристрою (ESP8266+BMP280). Параметри надсилаються у форматі #ключ#значення, де ключами є: temp (температура, T), pres (тиск, P), lx (освітленість, L), rssi (рівень сигналу Wi-Fi), vcc (напруга живлення). Формат завершено подвійною міткою ##, як вимагає протокол Narodmon.

Після передачі блоку даних виконується читання відповіді сервера методом client.readStringUntil('\r'), вміст якої виводиться до послідовного порту. З’єднання закривається викликом client.stop(). Значення lastNarodmonSendTime оновлюється значенням currentTime для подальшого контролю інтервалу.

Функція є синхронною, виконується у межах головного циклу програми та не використовує буферизацію чи асинхронні механізми. Такий підхід є достатнім у

межах обмежених ресурсів мікроконтролера ESP8266 та відповідає вимогам сервісу Narodmon щодо формату повідомлень і частоти запитів.

На платформі встановлено обмеження щодо частоти надсилання даних – мінімальний інтервал становить 5 хвилин. Тому у функціонал системи було вбудовано таймер, який відстежує час останньої передачі й визначає момент, коли дозволено надсилати нові дані. У разі помилки під час підключення до сервісу в консоль виводиться повідомлення про невдалу спробу. Якщо ж дані успішно надіслані – система також повідомляє про це через консоль.

Dweet.io є хмарним сервісом, орієнтованим на реалізацію механізмів Machine-to-Machine (M2M) комунікації в рамках Інтернету речей (IoT). Платформа забезпечує передачу телеметричних даних від сенсорних пристроїв до віддаленого серверного середовища за допомогою HTTP-запитів у форматі JSON. Кожне повідомлення (dweet) ідентифікується за унікальним іменем пристрою (thing name) і зберігається у тимчасовому сховищі з відкритим або обмеженим доступом.

Функціональність сервісу орієнтована на швидке підключення пристроїв без попередньої реєстрації або конфігурації облікових записів, що спрощує інтеграцію у прототипи та дослідницькі системи. Платформа підтримує RESTful-архітектуру та сумісна з більшістю програмованих мікроконтролерів і одноплатних комп’ютерів, які мають підтримку HTTP-клієнтів (наприклад, ESP32, Arduino з Ethernet/Wi-Fi-модулями, Raspberry Pi тощо).

Сервіс дозволяє зберігати та отримувати останні повідомлення, реалізовувати підписку на оновлення даних через сторонні сервіси (наприклад, Freeboard.io), а також забезпечує можливість інтеграції з власним хмарним сховищем або аналітичними інструментами. Це створює основу для побудови масштабованих систем збору та обробки даних у контексті моніторингу середовища, інженерних застосувань або наукових експериментів.

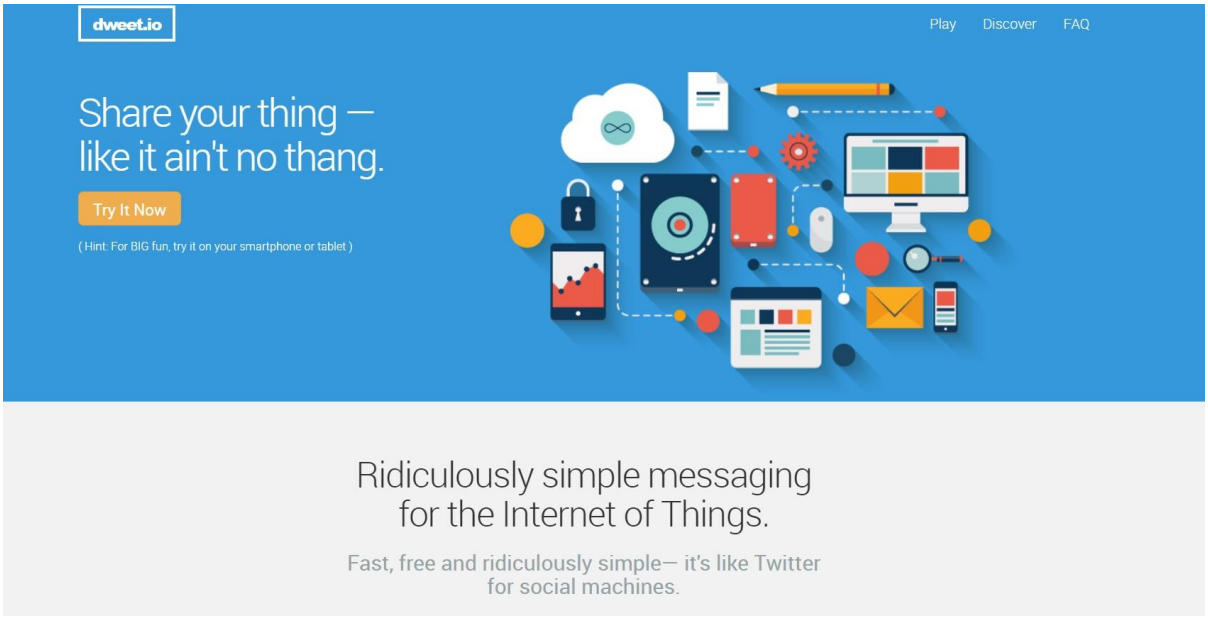


Рисунок 3.4 – Головна сторінка платформи Dweet.io

З цією метою – збереження даних та можливість їх подальшого використання – було прийнято рішення передавати інформацію на даний сервіс.

Функція dweetSend() виконує передачу сенсорних даних на платформу Dweet.io через HTTP-з'єднання, встановлене за допомогою клієнта WiFiClient. З'єднання ініціюється методом client.connect() до вузла dweet.io через порт 80. У разі відмови з'єднання функція завершує роботу, формуючи повідомлення про помилку в UART.

У разі успішного підключення формується HTTP-запит типу GET, у якому дані передаються у форматі query string як частина URL-адреси. Значення температури, тиску, освітленості, рівня сигналу Wi-Fi, напруги живлення та часу безперервної роботи пристрою формуються з відповідною точністю форматування та додаються до URI. Ідентифікатор пристрою задається як ESP8266_BMP280, що дозволяє однозначно ідентифікувати джерело даних на сервері.

Запит супроводжується обов'язковими HTTP-заголовками Host та Connection: close, що забезпечує відповідність специфікації HTTP/1.1 та дозволяє серверу коректно обробити запит і завершити сеанс з'єднання. Після надсилання запиту здійснюється зчитування відповіді до символу повернення каретки (\r) з

виведенням отриманих рядків у послідовний порт для діагностики. Після завершення прийому відповідь закривається методом client.stop().

Реалізація забезпечує передачу даних у відкритому текстовому форматі без використання додаткових протоколів чи бібліотек, що спрощує інтеграцію з REST-сервісами типу Dweet.io. Формування запиту виконується вручну відповідно до вимог серверного API, а отримані телеметричні значення можуть бути в подальшому оброблені або візуалізовані на стороні клієнта.

3.5 Ініціалізація функцій

Коректна ініціалізація функцій є ключовим чинником стабільної роботи всієї системи. Навіть за умови правильно написаного коду, помилки на етапі ініціалізації можуть призвести до некоректного функціонування пристрою. Тому важливо приділити особливу увагу правильній реалізації ініціалізації для забезпечення надійної та безперебійної роботи системи.

Функція setup() виконується один раз після запуску мікроконтролера ESP8266 і забезпечує ініціалізацію апаратних модулів та мережевих сервісів. Першим викликається Serial.begin(115200), що активує UART-інтерфейс з фіксованою швидкістю обміну 115200 біт/с для виводу діагностичної інформації.

UART (Universal Asynchronous Receiver-Transmitter) – це універсальний асинхронний приймач-передавач, що використовується для послідовного обміну даними між мікроконтролером і зовнішніми пристроями. Обмін інформацією відбувається через дві основні лінії: TX (transmit) – передача та RX (receive) – прийом. Комунікація в UART здійснюється без окремої лінії синхронізації, тому обидві сторони повинні мати узгоджену швидкість передачі (baud rate), наприклад, 115200 бод.

UART є базовим інструментом у налагодженні вбудованих систем. У випадку платформи ESP8266, цей інтерфейс дозволяє не лише здійснювати передачу логів через Serial Monitor у середовищі Arduino IDE, але й забезпечує завантаження

прошивки на мікроконтролер. UART широко використовується для діагностики, виводу показників датчиків, обробки команд від користувача та інтеграції з іншими мікросхемами або комп'ютером.

Ініціалізація сенсора BMP280 здійснюється викликом bmp.begin(0x76), де 0x76 – це вказана I²C-адреса пристрою. Після цього виконується connectWiFi(), яка реалізує підключення модуля до мережі Wi-Fi у режимі станції (WIFI_STA), з використанням SSID і пароля, закодованих у пам'яті або наданих динамічно. Після підключення до мережі функція setupTelegram() налаштовує TLS-сертифікати для захищеного обміну з Telegram Bot API, а також синхронізує час за допомогою NTP.

У консоль через UART виводиться MAC-адреса пристрою, отримана з WiFi.macAddress(), що дозволяє ідентифікувати вузол у локальній мережі. Далі здійснюється конфігурація обробників HTTP-запитів: маршрут "/" прив'язується до функції handleWebpage, яка повертає HTML-інтерфейс, тоді як шлях "/data" обробляється функцією handleData, яка формує JSON-відповідь із телеметричними значеннями. Завершальним етапом є запуск вбудованого веб-сервера командою server.begin(), що активує обробку вхідних HTTP-запитів на порту 80 згідно з протоколом HTTP/1.1.

Усі ці дії в сукупності забезпечують стартову підготовку системи до стабільної роботи.

Функція loop() реалізує головний цикл обробки подій, що виконується вбудованою системою після завершення ініціалізації. Її основним завданням є періодичне виконання ключових функціональних процедур: збирання сенсорних даних, перевірка стабільності мережевого підключення, обмін інформацією з зовнішніми сервісами та обробка HTTP-запитів і повідомлень Telegram API.

Інтервал між виконанням основних дій контролюється за допомогою функції millis(), яка відстежує час, що минув від моменту старту мікроконтролера. Після перевірки умови (millis() > lasttime + interval) викликається послідовність функцій: measure() для зчитування температури, тиску та живлення, luxt() – для вимірювання освітленості, checkWiFiConnection() – для перевірки актуального стану

підключення до мережі Wi-Fi, narodmonSend() – для відправлення даних на сервіс Narodmon через TCP, dweetSend() – для формування та надсилання HTTP GET-запиту на хост dweet.io.

У межах тієї ж умови здійснюється обробка вхідних повідомлень із Telegram через метод getUpdates(), що отримує нові повідомлення, і цикл while, у якому відбувається їх послідовна обробка функцією handleNewMessages() з оновленням стану bot.last_message_received. Завершення цього блоку супроводжується оновленням значення lasttime для повторного обчислення інтервалу виконання.

Незалежно від умов виконання таймеру, у кожній ітерації циклу loop() викликається server.handleClient(), який забезпечує обробку вхідних HTTP-запитів від клієнтів до вбудованого вебсервера на основі ESP8266WebServer. Цей виклик гарантує, що сервер залишається доступним для зовнішніх запитів навіть у періоди між циклами збирання та обробки сенсорних даних.

Таким чином, функція loop() реалізує основну логіку роботи автономної системи збору, аналізу та передавання даних у реальному часі, інтегрованої з мережею Internet of Things.

Збереження телеметричних даних є критичною функцією для реалізації ретроспективного аналізу, побудови моделей прогнозування та перевірки коректності роботи сенсорної системи. Проте мікроконтролери, що застосовуються у компактних вбудованих системах, як правило, мають обмежений обсяг вбудованої пам’яті. У досліджуваній конфігурації мікроконтролер оснащений флеш-пам’яттю обсягом 4 МБ, чого достатньо для зберігання прошивки, буферних структур та тимчасових даних, але недостатньо для повноцінної історичної архівації великих обсягів метеорологічної інформації.

З огляду на це, доцільним є використання зовнішніх хмарних платформ, які дозволяють централізовано зберігати, обробляти та візуалізувати зібрані дані. Зокрема, система підтримує інтеграцію з платформою Narodmon, що дозволяє не лише передавати поточні показники у відкритий доступ, а й реалізовувати логування у хмарному середовищі для подальшого аналізу. Платформа також

підтримує механізми маршрутизації даних на альтернативні сховища, включно з індивідуальними хмарними сервісами користувача.

У рамках реалізації системи було додатково передбачено збереження даних у хмарному сховищі Google Drive, яке слугує резервним каналом логування. Для кожної доби експлуатації створюється окремий файл у форматі CSV із фіксацією всіх зібраних параметрів. Обраний формат забезпечує сумісність з табличними процесорами, зокрема Google Sheets та Microsoft Excel, що спрощує візуальний перегляд, попередню фільтрацію та статистичну обробку зібраної інформації.

Отримані дані можна застосовувати для побудови графіків, проведення аналітики та формування прогнозів. Крім того, платформа надає вбудовані інструменти для візуалізації даних у вигляді графіків.

Завдяки функції збереження виміряних системою даних відкривається широкий простір для детального аналізу та прогнозування. Це значно розширює наші можливості й дозволяє виконувати багато корисних задач. Ми можемо використовувати ці дані для виявлення тенденцій, аналізу взаємозв’язків, прогнозування подій та прийняття обґрунтованих рішень.

3.6 Інструкція для авторизації програмного забезпечення

У портативному режимі метеостанція функціонує автономно, без підключення до ПК. Тому потрібно або створити нову точку доступу Wi-Fi, або перейменувати наявну відповідно до параметрів, заданих у системі.

Точка доступу з іменем мережі ESP8266 сконфігурована з використанням протоколу захисту WPA2-Personal, що реалізує шифрування трафіку за схемою з попередньо спільним ключем (PSK). Для автентифікації клієнтів задано пароль 12345678, який відповідає мінімальній довжині, визначеній стандартом IEEE 802.11i (8 символів).

SSID передається у відкритому режимі, тобто ідентифікатор мережі не приховується і доступний для виявлення пристроями у зоні покриття.

Ідентифікація точки доступу виконується як стандартна, що дозволяє автоматично підключатися сумісним пристроям, зокрема мікроконтролерам ESP8266, налаштованим у режимі WIFI_STA.

Задана конфігурація забезпечує базовий рівень безпеки з використанням сучасного стандарту шифрування AES (Advanced Encryption Standard) у межах WPA2, дозволяє виконувати обмін даними між мікроконтролером і мережею, а також слугує ізольованим середовищем для локального тестування IoT-систем без потреби в інфраструктурному Wi-Fi.

Після створення точки доступу відповідно до заданих параметрів, слід підключити живлення до системи. Це можна зробити за допомогою акумулятора, описаного раніше, або через кабель USB-A – microUSB, підключений до джерела живлення.

Рисунок 3.12 – Під’єднання мікроконтролера до джерела живлення за допомогою кабелю USB-A – microUSB.

Після чого мікроконтролер подасть сигнал світлодіодом про подачу живлення

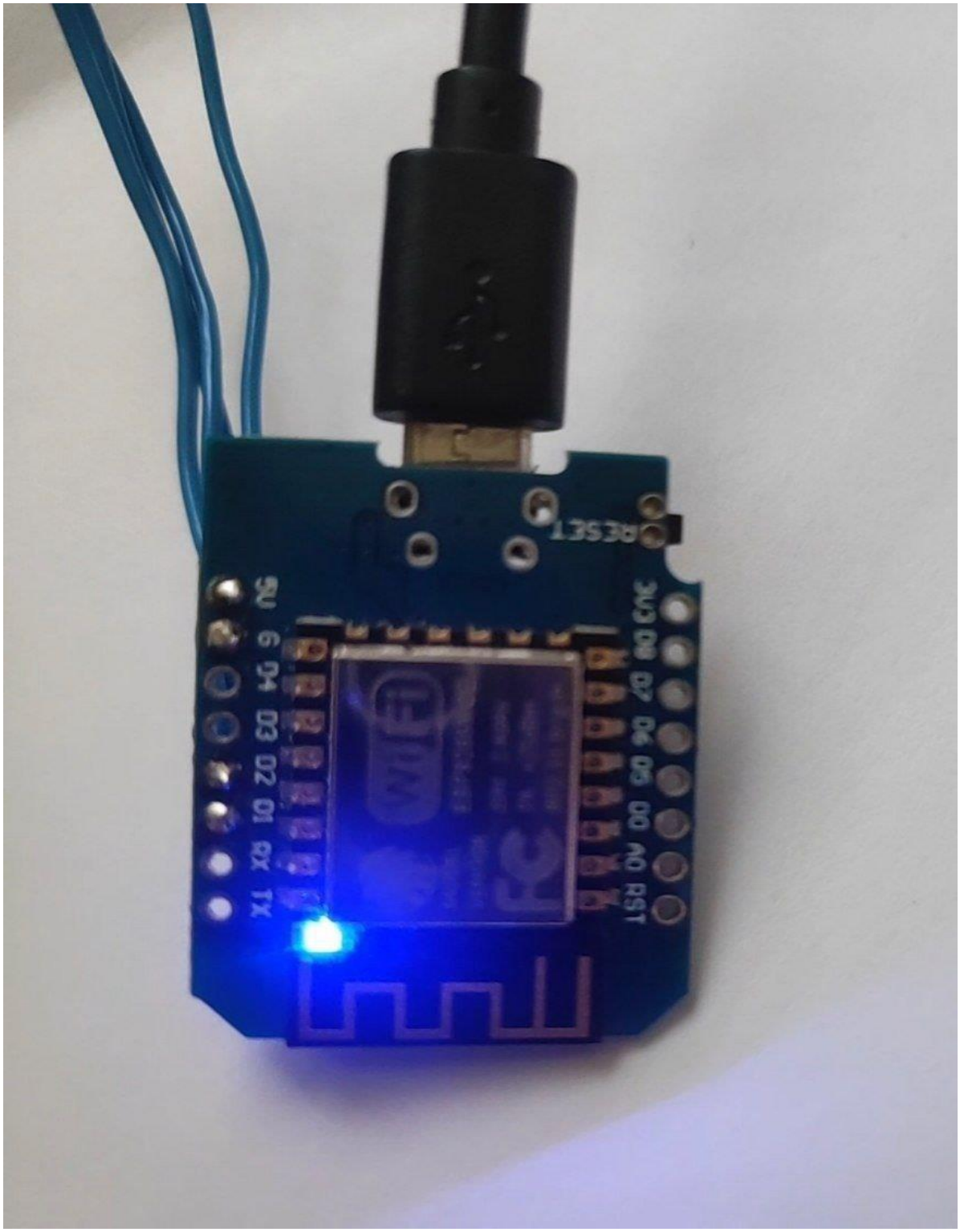


Рисунок 3.13 – Індикація світлодіода про початок роботи

Після підключення живлення потрібно зачекати приблизно 10–15 секунд – за цей час система автоматично під’єднається до вказаної мережі та почне свою роботу.

Telegram-бот з ідентифікатором @espbmp_bot зареєстрований під ім'ям ESP8266_BMP280. Бот призначений для інтеграції з мікроконтролером ESP8266 та сенсором BMP280 з метою дистанційного моніторингу метеопараметрів у режимі реального часу. Завдяки використанню Telegram Bot API, користувач може надсилати команди через інтерфейс месенджера й отримувати у відповідь значення температури, атмосферного тиску, освітленості, напруги живлення та рівня сигналу Wi-Fi. Ідентифікація бота здійснюється за унікальним іменем користувача у Telegram, що забезпечує однозначне адресування запитів та підтримку авторизованого доступу.

Інтерфейс Telegram-бота з ім'ям ESP8266_BMP280 реалізований через Telegram Bot API та активується після ініціалізації сесії користувачем через команду /start. Після запуску бот переходить у стан очікування вхідних команд, передбачених програмною логікою мікроконтролера ESP8266. Команди надсилаються у вигляді текстових запитів і обробляються бібліотекою UniversalTelegramBot.

Після активації доступні функції включають отримання значень температури, тиску, рівня освітленості, напруги живлення та RSSI. Взаємодія із ботом здійснюється у текстовому режимі, де кожна команда відповідає конкретному запиту до пристрою з подальшим формуванням структурованої відповіді у форматі текстового повідомлення. Це дозволяє здійснювати віддалений моніторинг стану системи без необхідності фізичного доступу до пристрою або окремого вебінтерфейсу.

Інтерфейс Telegram-бота ESP8266_BMP280 реалізовано у вигляді меню команд, кожна з яких викликає обробку відповідного запиту мікроконтролером. Серед доступних команд: /temp – запит на отримання значення температури з датчика BMP280; /pres – виклик для зчитування атмосферного тиску; /lux – зчитування рівня освітленості, що надходить із сенсора BH1750; /all – отримання повного набору параметрів, включаючи температуру, тиск, освітленість, напругу

живлення та RSSI; /help – виклик довідкового повідомлення з описом функціональності.

Усі команди обробляються програмно через бібліотеку UniversalTelegramBot, яка реалізує парсинг вхідного повідомлення та викликає відповідні функції. Відповідь надсилається у текстовому форматі без затримки, що забезпечує взаємодію в режимі реального часу. Такий підхід дозволяє здійснювати контроль параметрів системи дистанційно з використанням стандартного Telegram-клієнта.

У ході тестування функціональності Telegram-бота, реалізованого на базі мікроконтролера ESP8266 із підключеним сенсорним модулем BMP280, було здійснено обробку командного запиту користувача в режимі реального часу. Після отримання команди /temp бот сформував відповідь із поточним значенням температури, виміряної з роздільною здатністю до двох десяткових знаків. Передане значення становило 24.35 °C, що відповідає очікуваним показникам при кімнатній температурі.

Обробка команди /pres ініціювала запит до того самого сенсора з використанням методу readPressure(), після чого результат було перетворено з Паскалів у міліметри ртутного стовпчика (ділення на коефіцієнт 133.322). У відповідь користувач отримав значення 748.79 мм рт. ст., що свідчить про відсутність суттєвих барометричних змін.

Після надсилання команди /lux система зчитала дані з фотометричного сенсора BH1750 через інтерфейс I²C, ініціалізований у режимі CONTINUOUS_HIGH_RES_MODE. Отриманий результат 5.83 люкс було відформатовано у вигляді текстового повідомлення та повернуто користувачу. Такий рівень освітленості є типовим для середовищ із незначним фоновим освітленням або сутінковими умовами.

Передача даних здійснювалась через Telegram Bot API безпосередньо з прошивки мікроконтролера. Формат відповіді має текстову структуру, оптимізовану для сприйняття кінцевим користувачем. Застосування бібліотеки

UniversalTelegramBot забезпечило обробку вхідних повідомлень та автоматичну генерацію відповідей без потреби в проміжних серверах або проксі-рішень.

У результаті обробки команди /all, надісланої через Telegram-бот, реалізований на мікроконтролері ESP8266, система формує та передає агрегований набір телеметричних параметрів, зчитаних у режимі реального часу з периферійних сенсорів. До складу відповіді входить температура повітря, виміряна сенсором BMP280, значення якої на момент виклику становило 24.32 °C. Атмосферний тиск, отриманий з того ж модуля, після перетворення з Па у мм рт. ст. дорівнював 748.83 мм рт. ст. Яскравість освітлення, зчитана з фотометричного сенсора BH1750, становила 5.83 люкс.

Додатково виводиться рівень напруги живлення, визначений за допомогою методу ESP.getVcc(), що в даному випадку становив 2.99 В. Рівень сигналу бездротової мережі, зчитаний функцією WiFi.RSSI(), дорівнював –43 dBm. Усі значення сформовано у форматі текстового повідомлення з фіксованою структурою для подальшої обробки або моніторингу.

У відповідь на команду /help Telegram-бот передає інструкцію щодо доступних команд управління, реалізованих у програмному інтерфейсі системи. Перелік включає: /temp – зчитування температури, /pres – тиску, /lux – освітленості, та /all – запит повного набору даних. Команди обробляються через Telegram Bot API із застосуванням бібліотеки UniversalTelegramBot, що дозволяє асинхронну взаємодію з користувачем у форматі запит–відповідь із мінімальною затримкою.

Для розгортання проєкту у стаціонарному режимі необхідно встановити інтегроване середовище розробки Arduino IDE (наведено приклад версії 2.2.2). Після запуску середовища автоматично створюється скетч із базовою структурою функцій setup() і loop(), що відповідає архітектурним вимогам платформі Arduino.

У центральному вікні відображено редактор коду з можливістю інтерактивного написання та компіляції програм. У нижній частині інтерфейсу виводиться попередження про відсутність активного з'єднання з апаратною платформою: «Не підключено. Оберіть плату та порт для автоматичного

підключення». Для забезпечення коректної роботи з мікроконтролером необхідно вручну вибрати відповідний тип плати (наприклад, LOLIN(WEMOS) D1 mini) і COM-порт, до якого підключено пристрій.

Також рекомендовано встановити швидкість обміну даними в моніторі порту на значення 115200 бод, що є стандартом для мікроконтролерів серії ESP8266 під час налагодження через UART. Це дозволить отримувати діагностичні повідомлення із серійного інтерфейсу в режимі реального часу.

Далі слід підключити метеостанцію до персонального комп'ютера за допомогою кабелю USB-A – microUSB та здійснити відповідні налаштування в середовищі розробки.

Для забезпечення можливості завантаження програмного коду на мікроконтролер необхідно виконати апаратне підключення пристрою до персонального комп'ютера за допомогою кабелю типу USB-A – microUSB. Після встановлення фізичного з'єднання здійснюється конфігурація програмного середовища Arduino IDE.

У відповідному діалоговому вікні користувач повинен вручну обрати модель плати, сумісну з використовуваним пристроєм. У межах даного проєкту було вибрано плату LOLIN(WEMOS) D1 mini (clone), яка є апаратною реалізацією мікроконтролера ESP8266. Далі виконується вибір COM-порту, що відповідає активному USB-інтерфейсу комп'ютера, до якого підключено метеостанцію – у прикладі вказано COM7.

Зазначена конфігурація є обов'язковою умовою для успішної компіляції скетчів, їх подальшого завантаження в мікроконтролер, а також використання монітору порту для відлагодження системи через UART-інтерфейс.

У випадку, коли параметри точки доступу, задані у програмному забезпеченні мікроконтролера, відповідають реальним конфігураціям бездротової мережі, на UART-інтерфейс виводиться послідовність повідомлень, що підтверджують встановлення з'єднання з Wi-Fi мережею. Зокрема, фіксується призначена локальна IP-адреса, ініціюється підключення до Telegram Bot API із

використанням TLS через WiFiClientSecure, після чого відображається MAC-адреса пристрою.

Подальший блок виводу містить результат зчитування параметрів з датчиків: температури навколишнього середовища (BMP280), атмосферного тиску (із конвертацією у мм рт. ст.), напруги живлення мікроконтролера (через ESP.getVcc()), а також рівня сигналу Wi-Fi (WiFi.RSSI()). Освітленість визначається за допомогою фотометра BH1750. У завершальній частині передається статус взаємодії з віддаленим сервісом Dweet.io, що свідчить про успішну передачу даних у хмарне середовище. Вивід цих даних вказує на завершення ініціалізації та готовність пристрою до виконання основного циклу задач у безперервному режимі.

У випадку, коли параметри SSID та пароля точки доступу, збережені в системі, не дозволяють встановити з’єднання, модуль автоматично ініціює процедуру активного сканування бездротового середовища. Після завершення пошуку система формує перелік виявлених Wi-Fi мереж, доступних для підключення, та очікує введення SSID користувачем через UART-інтерфейс.

Цей механізм реалізовано із використанням функції WiFi.scanNetworks(), що повертає кількість виявлених мереж, а доступ до SSID кожної з них здійснюється через WiFi.SSID(i). Після введення користувачем імені мережі та пароля система виконує підключення за допомогою методу WiFi.begin(). Такий підхід дозволяє змінювати мережеві параметри динамічно без необхідності повторного програмування мікроконтролера, що підвищує гнучкість розгортання пристрою в різних середовищах.

У разі невдалого підключення до заздалегідь визначеної Wi-Fi мережі, модуль ESP8266 ініціює процедуру сканування доступних бездротових мереж, використовуючи функцію WiFi.scanNetworks(). Після завершення сканування в UART-консолі відображається кількість виявлених SSID, а також перелік ідентифікованих мереж.

Користувач отримує можливість вручну ввести назву мережі (SSID), до якої необхідно встановити підключення. Уведення здійснюється через серійний монітор. Після верифікації наявності введеного SSID серед доступних точок, мікроконтролер виконує команду WiFi.begin(), ініціюючи підключення до вибраної мережі.

Реалізований підхід дозволяє адаптуватися до зміни мережевих умов без необхідності модифікації прошивки. Це особливо актуально для автономних IoT-систем, де важливо забезпечити збереження функціональності при втраті основного з’єднання. У результаті підвищується гнучкість і надійність процесу конфігурації мережевого інтерфейсу.

У разі неможливості встановлення з'єднання з попередньо визначеною Wi-Fi мережею мікроконтролер автоматично ініціює процедуру сканування доступних точок доступу з використанням функції WiFi.scanNetworks(). Після завершення сканування виводиться перелік SSID знайдених мереж, що дозволяє оператору вручну обрати альтернативну мережу для підключення.

Інтерфейс введення у моніторі порту запитує від користувача назву SSID та відповідний пароль. Введені дані використовуються для встановлення нового підключення за допомогою WiFi.begin(ssid, password). Якщо введене SSID не збігається з жодною з доступних мереж, процедура повторюється до успішного підключення або вручну зупиняється користувачем.

Цей механізм забезпечує динамічну конфігурацію параметрів мережевого підключення без потреби в перепрошиванні пристрою, що особливо доцільно для мікропроцесорних систем, експлуатованих у змінному середовищі з непостійною конфігурацією точок доступу.

Після введення обраного SSID та відповідного пароля, модуль ESP8266 здійснює підключення до зазначеної Wi-Fi мережі. У випадку успішної аутентифікації, виводиться повідомлення про встановлення з'єднання, а також локальна IP-адреса, присвоєна пристрою DHCP-сервером маршрутизатора.

Паралельно ініціалізується TLS-з'єднання з Telegram-сервером, після чого виконується авторизація Telegram-бота. На цьому етапі підтверджується успішне з’єднання з Telegram API, що дозволяє подальший обмін повідомленнями між ботом і користувачем.

Далі здійснюється зчитування сенсорних даних з підключених модулів: температура (T = 28.50 °C), атмосферний тиск (P = 748.68 мм рт. ст.), освітленість (L = 8.33 лк), напруга живлення (VCC = 2.993 В), рівень потужності прийнятого сигналу (RSSI = –55 dBm). Зібрана інформація публікується на платформі Narodmon.org шляхом формування HTTP-запиту через TCP-клієнт. Після отримання відповіді від сервера Narodmon виводиться повідомлення про успішну передачу. Аналогічним чином підтверджується успішна інтеграція з хмарним сервісом dweet.io.

У разі введення користувачем SSID і пароля до Wi-Fi-мережі, підключення ініціалізується через функцію WiFi.begin(). Якщо протягом заданого інтервалу часу (тайм-ауту), визначеного в мілісекундах, не вдалося встановити з'єднання з точкою доступу, система виводить повідомлення про завершення очікування.

Це сигналізує про виникнення помилки на етапі аутентифікації або асоціації з мережею. Причиною може бути некоректне введення облікових даних, слабкий рівень сигналу або фільтрація MAC-адрес на стороні маршрутизатора. Після фіксації невдачі система автоматично повторює спробу підключення відповідно до реалізованого циклу обробки підключення, що забезпечує безперервність роботи пристрою в автономному середовищі.

Протягом кожного циклу виконання функції передачі даних через API платформи dweet.io здійснюється зчитування значень ключових метеорологічних і технічних параметрів. Вимірювана температура повітря (T) коливалася в межах від 26.63 °C до 27.46 °C, атмосферний тиск (P) – у межах 748.62–748.68 мм рт. ст. Напруга живлення мікроконтролера (VCC) залишалась стабільною – близько 2.993–2.995 В. Рівень сигналу Wi-Fi (RSSI) коливався від –58 до –46 dBm, що відповідає прийнятному діапазону якості з’єднання для ESP8266.

Рівень освітленості, виміряний фотодатчиком BH1750 (L), варіювався в межах 31.67–43.33 люкс. Після кожного вимірювання дані передавались на сервер dweet.io, про що свідчить повідомлення dweet ok!, яке інтерпретується як підтвердження успішної HTTP-транзакції з боку сервера.

Циклічний режим роботи системи демонструє стабільність вимірювань та коректність функціонування процедури відправлення даних у форматі GET-запиту до віддаленого серверного інтерфейсу.

3.7 Висновок до розділу

У розділі реалізовано архітектурно-функціональну модель програмно-апаратного комплексу для збору та обробки метеорологічних даних на базі мікроконтролера ESP8266. Здійснено апаратну інтеграцію сенсорних елементів (DHT22, BME280, FC-37), які забезпечують вимірювання температури, відносної вологості, атмосферного тиску та фіксацію наявності опадів. Усі компоненти обрано з урахуванням сумісності за напругою живлення, типом інтерфейсу та метрологічними характеристиками.

Розроблено програмне забезпечення, яке забезпечує зчитування даних із сенсорів, обробку вимірювань, формування телеметричних повідомлень у форматі JSON та їх виведення на локальний дисплей (OLED SSD1306) або передавання на хмарну платформу (ThingSpeak). Забезпечено підтримку передачі даних через Wi-Fi з використанням протоколу HTTP, що дає змогу здійснювати віддалений моніторинг параметрів у реальному часі.

Підготовлено конфігурацію бази даних для накопичення результатів вимірювання, що створює основу для подальшої аналітики та реалізації алгоритмів короткострокового прогнозування на основі часових рядів. Визначено принципи модульної побудови системи, які дозволяють масштабувати її функціональні можливості, зокрема за рахунок додавання нових сенсорів або каналів передавання даних.

Отримані результати підтверджують функціональну придатність обраної апаратно-програмної платформи для побудови розподіленої кіберфізичної системи моніторингу довкілля з можливістю інтеграції у більш складні системи класу IoT та UAV.

ВИСНОВКИ

У межах кваліфікаційної роботи виконано повний цикл розробки кіберфізичної системи для прогнозування опадів, реалізованої на базі мікроконтролера ESP8266. Проведено системний аналіз предметної області, що охоплює структуру, функціональні характеристики та сферу застосування метеорологічних систем моніторингу. Обґрунтовано вибір архітектури апаратно-програмного комплексу з урахуванням вимог до енергоспоживання, точності вимірювань, масштабованості та сумісності з інтерфейсами IoT.

У ході дослідження реалізовано апаратну частину системи з використанням сенсорних модулів DHT22, BME280 та FC-37, які забезпечують вимірювання ключових метеорологічних параметрів: температури, вологості, атмосферного тиску та наявності опадів. Центральний вузол побудовано на платформі ESP8266, що виконує обробку отриманих даних, локальну індикацію результатів на OLED-дисплеї та передавання інформації на зовнішні сервіси за допомогою протоколу HTTP.

Програмне забезпечення системи реалізовано у середовищі Arduino IDE з використанням відкритих бібліотек. Забезпечено модульність коду, що дозволяє адаптувати функціональність системи до різних експлуатаційних сценаріїв. Підготовлено структуру бази даних для архівування результатів вимірювань з подальшою можливістю їх аналітичної обробки та візуалізації на платформі ThingSpeak.

Проведене тестування підтвердило працездатність системи, відповідність заданим метрологічним характеристикам та стабільність роботи в автономному режимі. Запропоноване рішення може бути використане як основа для розгортання розподілених метеомереж, а також для інтеграції з безпілотними платформами в контексті побудови розвідувальних або моніторингових кіберфізичних систем.

Отримані результати демонструють практичну доцільність застосування недорогих мікроконтролерів з підтримкою бездротових інтерфейсів у задачах екологічного моніторингу та прогнозування стану навколишнього середовища

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Hritam Sarkar. Home-Based and Portable Smart Weather Station. *Int. Jour. Of Research*. 2024. URL: https://www.irejournals.com/formatedpaper/1708826.pdf (дата звернення: 25.05.2025).
2. Tamal Sarkar. Design of A Cost-Effective Weather Station with Rain Forecast, 2023. URL: https://www.researchgate.net/publication/380226214_Design_of_A_Cost-Effective_Weather_Station_with_Rain_Forecast (дата звернення: 17.04.2025).
3. Solar Powered WiFi Weather Station V3.0 на Instructables із BMP280/BME280. URL: https://www.instructables.com/Solar-Powered-WiFi-Weather-Station-V30 (дата звернення: 26.04.2025).
4. WiFi Weather Station Prediction using ML (IJNRD, 2023). URL: https://www.ijnrd.org/papers/IJNRD2311343.pdf (дата звернення: 26.04.2025).
5. IoT-enabled Smart Weather Stations (MJSAT, 2023)
6. IoT-Based Weather Analysis System Using ESP8266 (IJRASET, 2023). URL: https://www.ijraset.com/research-paper/iot-based-weather-analysis-system-using-esp8266 (дата звернення: 19.04.2025).
7. A Development IoT Based Real-Time Weather Monitoring System, ESP32/BMP280/DHT11 (InforKUM, 2025)
8. ESP-Based Weather Station. URL: https://www.irejournals.com/formatedpaper/1708826.pdf (дата звернення: 01.04.2025).
9. Predicting Weather Conditions by the IoT Platform, Atlantis Press, 2023. URL: https://www.atlantis-press.com/article/125984570.pdf (дата звернення: 23.05.2025).
10. IoT-enabled Smart Weather Stations: Innovations, ResearchGate, 2023. URL: https://www.researchgate.net/publication/379878763_IoT-enabled_Smart_Weather_Stations_Innovations_Challenges_and_Future_Directions
11. IoT Live Weather Monitoring System Using NodeMCU ESP8266, IRJMETS, 2022. URL:

https://www.irjmets.com/uploadedfiles/paper/issue_5_may_2022/24819/final/fin_irjmets1653918150.pdf (дата звернення: 29.04.2025).

12. IoT Based Live Weather Station Monitoring System. *Materials Science & Tech*. 2023. URL: https://materialsciencetech.com/mst/uploads/2024-42651.pdf (дата звернення: 30.04.2025).

13. Environmental Sensor with Arduino and BME280, *Luis Llamas* tutorial. URL: https://www.luisllamas.es/en/environmental-sensor-arduino-bme280 (дата звернення: 13.05.2025).

14. Датчик атмосферного тиску, температури BMP180. URL: https://ardushop.in.ua/arduino/atmospheric-pressure-temperature-sensor-bmp180 (дата звернення: 02.04.2025).

15. An IoT based weather station using an embedded system, ResearchGate, 2022. URL: https://www.researchgate.net/publication/364628364_An_IoT_based_weather_station_using_an_embedded_system (дата звернення: 19.05.2025).

16. Room Monitoring Uses ESP-12E Based DHT22 and BH1750, *Journal UMY*, 2021. URL: https://journal.umy.ac.id/index.php/jrc/article/view/11023 (дата звернення: 10.05.2025).

17. Prototyping Low-Cost Automatic Weather Stations, *ScienceDirect*, 2022. URL: https://www.sciencedirect.com/science/article/pii/S2352864822000931 (дата звернення: 23.05.2025).

18. DeepRain: ConvLSTM for Precipitation Prediction, *arXiv*. URL: https://arxiv.org/abs/1711.02316 (дата звернення: 13.05.2025).

19. Deep CNN Model for Improving WRF Forecasts, *arXiv* 2020. URL: https://arxiv.org/abs/2008.06489 (дата звернення: 13.05.2025).

20. Raspberry Pi + Arduino Uno as Basic Meteorological Station, *arXiv* https://arxiv.org/abs/1711.09750 (дата звернення: 26.03.2025).

21. Skillful Precipitation Nowcasting, *arXiv* 2021. URL: https://arxiv.org/abs/2104.00954

22. PERSIANN satellite precipitation dataset, Wikipedia. URL: https://en.wikipedia.org/wiki/PERSIANN

23. Espressif Systems, ESP8266 Technical Reference Manual. URL: https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf (дата звернення: 26.05.2025).

24. Internet of Things: Architectures, Protocols, and Applications. URL: https://www.researchgate.net/publication/312957467_Internet_of_Things_Architectures_Protocols_and_Applications (дата звернення: 10.04.2025).

25. "Solar Powered WiFi Weather Station V3.0" Instructables BMP280/BME280. URL: https://github.com/2black0/Weather-Station-ESP8266

26. Arduino-форум: "BMP280 and Weather Station" (I2C з ESP8266). URL: https://forum.arduino.cc/t/bmp280-and-weather-station/375446 (дата звернення: 17.04.2025).

27. "ESP8266, BMP280, MQTT Weather Station" на Instructables. URL: https://www.instructables.com/ESP8266-BMP280-MQTT-Weather-Station (дата звернення: 28.05.2025).

28. Kodak GitHub «esp8266-weather-station-oled-bmp280» (з SSD1306 OLED). URL: https://github.com/ubiclub/esp8266-weather-station-oled-bmp280 (дата звернення: 28.05.2025).

29. Weather Station With ESP8266, BMP180, DHT22. URL: https://www.instructables.com/Weather-Station-With-ArduinoGenuino-ESP8266-BMP180 (дата звернення: 23.05.2025).

30. GitHub: 2black0/esp8266-weather-station (ThingSpeak + Telegram). URL: https://github.com/2black0/Weather-Station-ESP8266 (дата звернення: 30.05.2025).

31. Telegram: Request ESP32/ESP8266 Sensor Readings (Arduino IDE). URL: https://randomnerdtutorials.com/telegram-request-esp32-esp8266-nodemcu-sensor-readings (дата звернення: 28.05.2025).

32. DIY Project – Monitor and Control Weather Station with Telegram App. URL: https://blog.make2explore.com/diy-project-monitor-and-control-weather-station-with-telegram-app (дата звернення: 17.05.2025).

33. Interface DHT11 Module With Arduino. URL: https://lastminuteengineers.com/dht11-module-arduino-tutorial/ (дата звернення: 01.04.2025).

34. MathWorks ThingSpeak: ESP8266 & BME280 & BH1750. URL: https://thingspeak.mathworks.com/channels/114043 (дата звернення: 21.05.2025).

35. Датчик температури вологості та тиску Барометр BME280. URL: https://rozetka.com.ua/ua/395164584/p395164584/?gad_source=1&gad_campaignid=20086286791&gbraid=0AAAADlXhI4WroazHMjAHvyrEozz97Ngr&gclid=CjwKCAjwx8nCBhAwEiwA_z__06KcgO2Pli8r14w6w60SemkTweTk-p9x7c4KKqiqfyNqlW4yXduS0RoCGecQAvD_BwE (дата звернення: 01.04.2025).

36. "Easy ThingSpeak Home Weather Station". URL: https://www.instructables.com/Easy-Thingspeak-Home-Weather-Station-Hygrometer-Wi (дата звернення: 17.05.2025).

37. IToHI: Building ESP8266 Weather Station with BME280&MQTT & DeepSleep. URL: https://itohi.com/blog/building-esp8266-weather-station-part-i (дата звернення: 27.03.2025).

38. R. Cortés León et al., "Raspberry Pi and Arduino UNO Working together as a Basic Meteorological Station". *arXiv*, Nov 2017.

39. FC-37 модуль – датчик вологоті. URL: https://gurtivnya-elektryky.com.ua/ua/p1580432106-modul-datchik-vlazhnostidozhdya.html (дата звернення: 05.04.2025).

40. Adeagbo, "IOT Based Environment Monitoring System Using", *arXiv*, May 2024. URL: https://arxiv.org/abs/2405.14047 (дата звернення: 10.05.2025).

41. "IoT-enabled Smart Weather Stations", MJSAT, PDF (BME280, BH1750)

42. The most popular HTML, CSS, and JS library in the world. URL: https://getbootstrap.com/docs/4.6/getting-started/introduction/ (дата звернення: 11.04.2025).

43. A H M Jakaria et al., “Smart Weather Forecasting Using Machine Learning: A Case Study in Tennessee,” *arXiv:2008.10789*, Aug 2020. URL: https://arxiv.org/abs/2008.10789 (дата звернення: 23.05.2025).

44. Theodore Karachalios et al., “Arduino Sensor Integrated Drone for Weather Indices: A Prototype for Pre-flight Preparation,” *arXiv:2106.16083*, Jun 2021. URL: https://arxiv.org/abs/2106.16083 (дата звернення: 29.03.2025).

45. Adam Giammarese et al., “Tree-based Learning for High-Fidelity Prediction of Chaos,” *arXiv:2403.13836*, Mar 2024 https://arxiv.org/pdf/2403.13836 (дата звернення: 10.05.2025).

46. (MDPI) “Low-Cost Automatic Weather Stations in the Internet of Things,” *Information*, vol. 12, no. 4, 2021. URL: https://www.mdpi.com/2078-2489/12/4/146 (дата звернення: 16.05.2025).

47. Suman Kumar Das & Pujyasmita Nayak, “Integration of IoT-AI powered local weather forecasting: A Game-Changer for Agriculture”, *arXiv:2501.14754*, Jan 2025 https://arxiv.org/abs/2501.14754 (дата звернення: 16.05.2025).

48. Julian Grinkevic, “IoT Weather Station for Smart Buildings using LoRaWAN Transmission”, BSc thesis, Apr 2022. URL: https://www.researchgate.net/publication/386134759_IoT_Weather_Station_for_Smart_Buildings_using_LoRaWAN_Transmission (дата звернення: 01.04.2025).

49. Avines Panneer Selvam & Safaa N. S. Al-Humairi, “The Impact of IoT and Sensor Integration on Real-Time Weather Monitoring Systems: A Systematic Review”, 2023. URL: https://www.researchgate.net/publication/375567504_The_Impact_of_IoT_and_Sensor_Integration_on_Real-Time_Weather_Monitoring_Systems_A_Systematic_Review (дата звернення: 19.05.2025).

50. TFT LCD дисплей 2.4" 320x240 модуль ST7789V SPI інтерфейс. URL: https://rozetka.com.ua/ua/515950719/p515950719/?gad_source=1&gad_campaignid=20086286791&gbraid=0AAAADlXhI7_68KvTYIzJmE-EZ52IHrIa&gclid=CjwKCAjwx8nCBhAwEiwA_z__030dpbXY7Kl4gCY83HKQ_THrzcD0P098LFm31YuejJeW9XT_WZn8wxoC6I8QAvD_BwE (дата звернення: 13.05.2025).

51. Anita B. Agarwal et al., “Spatially-resolved hyperlocal weather prediction and anomaly detection using IoT sensor networks and machine learning techniques”, *arXiv:2310.11001*, Oct 2023

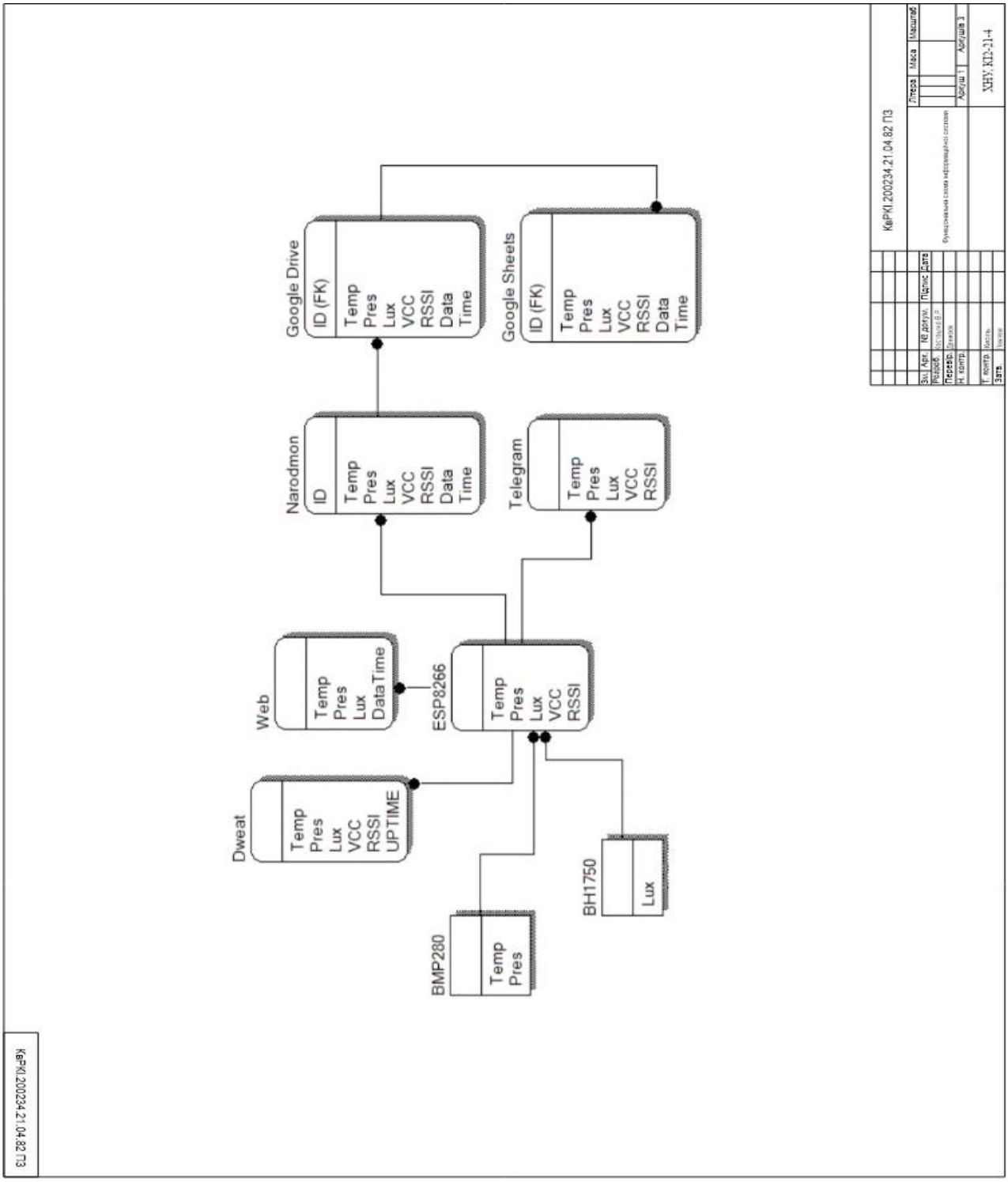
52. How to Build a Weather Station with ESP8266 and OLED Display: Fetch Real-Time Weather Data. URL: https://www.circuitmagic.com/robotics/how-to-build-a-weather-station-with-esp8266-and-oled-display-fetch-real-time-weather-data/ (дата звернення: 27.03.2025).

53. ESP8266 NodeMCU with Rain Sensor and DHT11 Weather Station. URL: https://docs.cirkitdesigner.com/project/published/aa90b903-5000-441d-92da-19392db18cf7/esp8266-nodemcu-with-rain-sensor-and-dht11-weather-station (дата звернення: 23.05.2025).

Додаток А

(обов’язковий)

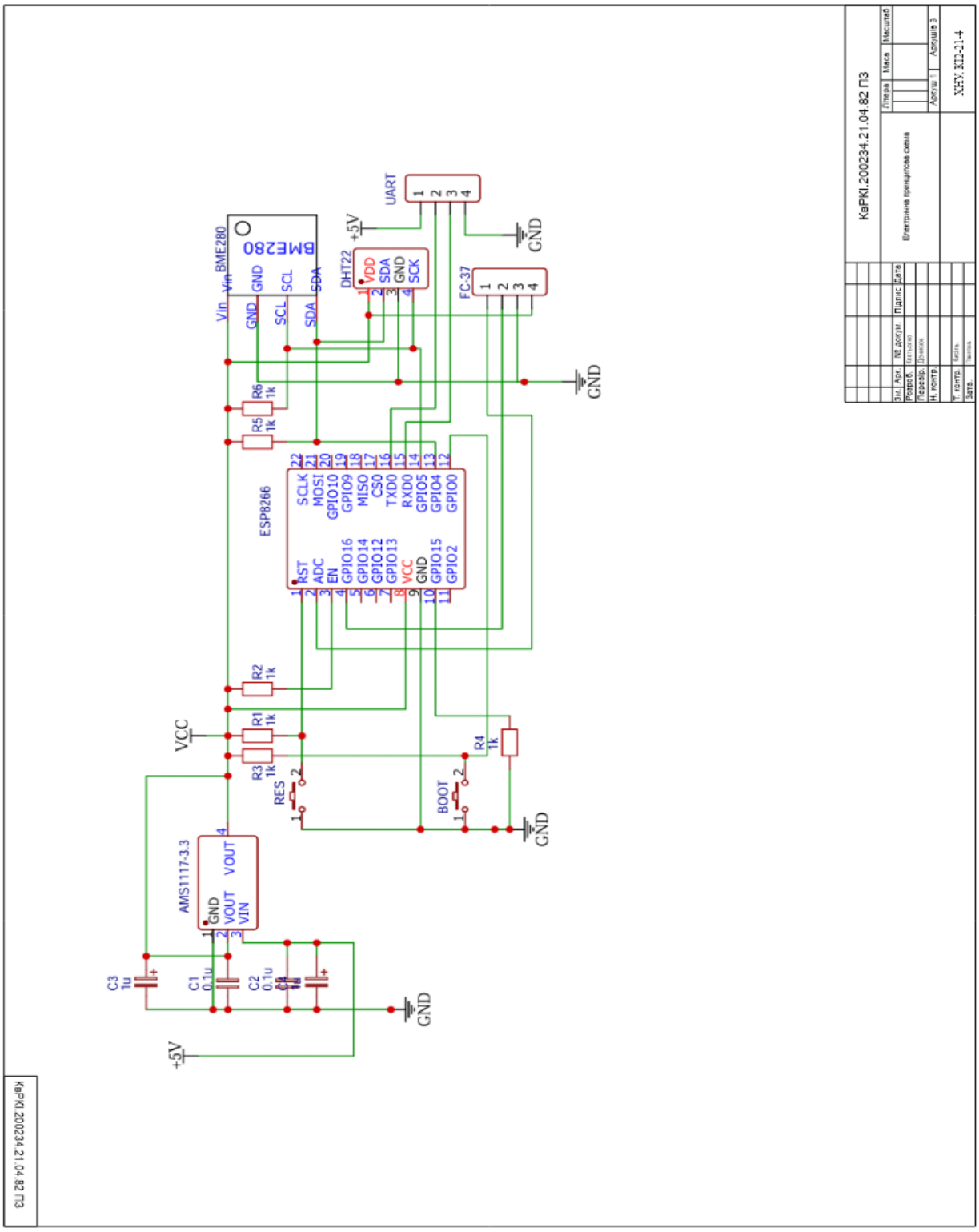
ФУНКЦІОНАЛЬНА СХЕМА ІНФОРМАЦІЙНОЇ СИСТЕМИ



Додаток Б

(обов’язковий)

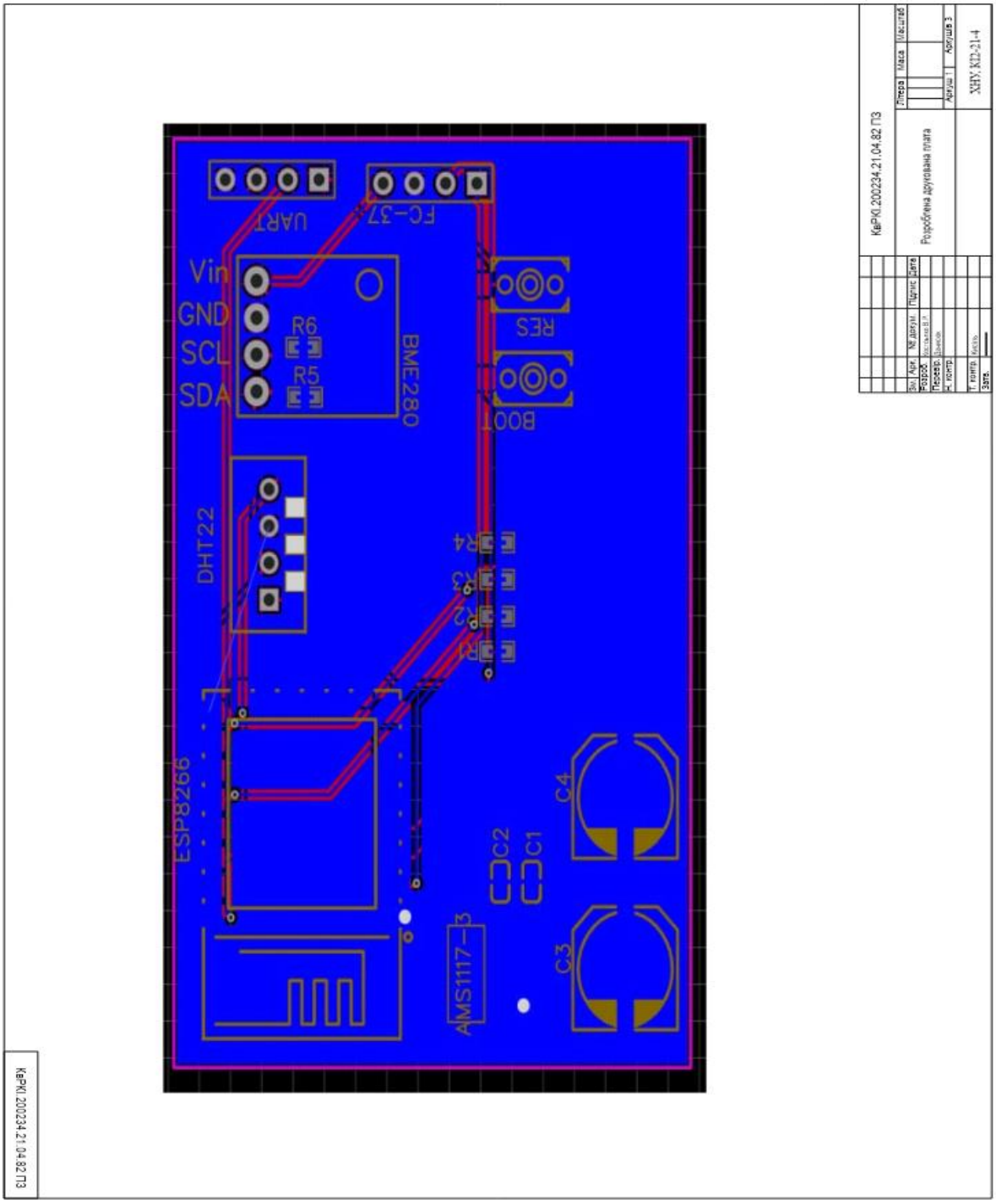
ЕЛЕКТРИЧНА ПРИНЦИПОВА СХЕМА



Додаток В

(обов’язковий)

РОЗРОБЛЕНА ДРУКОВАНА ПЛАТА



Додаток Г

(обов'язковий)

ПРОГРАМНИЙ КОД

```
#include <ESP8266WiFi.h>
#include <Wire.h>
#include "WiFiClient.h"
#include "ESP8266WebServer.h"
#include "ESP8266HTTPClient.h"
#include "ESP8266mDNS.h"
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP280.h>
#include <BH1750.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

ADC_MODE(ADC_VCC);
Adafruit_BMP280 bmp; // I2C
BH1750 lightMeter;
WiFiClient client;
ESP8266WebServer server(80);

#define host "eu.narodmon.com"
#define httpPort 8283
#define dweet "dweet.io"
#define dweetPort 80
#define interval 5000
   #define         BOT_TOKEN
"6561845762:AAFul7SqNUqrJwoxJeY8SibZ9rGgM3PZ4is"

X509List cert(TELEGRAM_CERTIFICATE_ROOT);
WiFiClientSecure secured_client;
UniversalTelegramBot bot(BOT_TOKEN, secured_client);
```

```
float vcc, T, P, L; int rssi; unsigned long lasttime; unsigned long lastNarodmonSendTime = 0; const unsigned long narodmonSendInterval = 5 * 60 * 1000; // 5 хвилин у
мілісекундах

String generateWebpage() {
  String webpage = "<!DOCTYPE html>"; webpage += "<html>"; webpage += "<head>";
    webpage += "<link rel=\"stylesheet\"
href=\"https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css\">"; webpage += "<style>"; webpage += "body {"; webpage += " font-family: Arial, Helvetica, sans-serif;"; webpage += " background-color: #f2f2f2;"; webpage += "}"; webpage += ".card {"; webpage += " background-color: white;"; webpage += " border-radius: 10px;"; webpage += " box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2);"; webpage += " transition: 0.3s;"; webpage += " width: 300px;"; webpage += " margin: auto;"; webpage += " padding: 20px;"; webpage += "}";
  webpage += ".container {";
  webpage += " text-align: left;"; webpage += "}";
  webpage += ".icon {"; webpage += " float: left;"; webpage += " margin-right: 10px;"; webpage += "}"; webpage += "</style>"; webpage += "<script>"; webpage += "function updateTime() {"; webpage += " var now = new Date();"; webpage += " var dateElement = document.getElementById('date-time');"; webpage += " dateElement.innerHTML = now.toLocaleString();"; webpage += "}"; webpage += "setInterval(updateTime, 1000);"; // Оновлюємо кожну
  секунду webpage += "function fetchData() {"; webpage += " fetch('/data')"; webpage += " .then(response => response.json())"; webpage += " .then(data => {";
    webpage += " document.getElementById('temp').innerText = data.temperature.toFixed(2) + ' °C';";
    webpage += " document.getElementById('pres').innerText =
```

```
    data.pressure.toFixed(2) + ' mmHg';"; webpage += "
document.getElementById('lux').innerText =
    data.light.toFixed(2) + ' lx';"; webpage += " });"; webpage
+= "}"; webpage += "setInterval(fetchData, 1000);"; // Оновлюємо
кожну секунду webpage += "</script>"; webpage += "</head>";
    webpage += "<body onload=\"updateTime(); fetchData();\">";
webpage += "<div class=\"card\">";
      webpage += " <div class=\"container\">"; webpage += "
<h4><b>Temperature</b></h4>"; webpage += " <div class=\"icon\"><i
class=\"fas fa-thermometer-
    half\"></i></div>"; webpage += " <span
id=\"temp\">Loading...</span><br>"; webpage += "
<h4><b>Pressure</b></h4>"; webpage += " <div class=\"icon\"><i
class=\"fas fa-tachometer-
    alt\"></i></div>"; webpage += " <span
id=\"pres\">Loading...</span><br>"; webpage += " <h4><b>Light
Intensity</b></h4>";
        webpage += " <div
       class=\"icon\"><i class=\"fas fa-
    lightbulb\"></i></div>"; webpage += " <span
id=\"lux\">Loading...</span><br>"; webpage += " <h4><b>Date and
Time</b></h4>"; webpage += " <div class=\"icon\"><i class=\"far
fa-calendar-
    alt\"></i></div>"; webpage += " <span id=\"date-
time\"></span>"; webpage += " </div>"; webpage += "</div>";
webpage += "</body>"; webpage += "</html>"; return webpage;
    }

    void handleData() {
      StaticJsonDocument<200> data;

    data["temperature"] = T;
    data["pressure"] = P; data["light"] = L;

      String jsonData; serializeJson(data, jsonData);
```

```
  server.send_P(200, "application/json", jsonData.c_str());
}

void handleWebpage() { server.send(200, "text/html", generateWebpage());
}

void connectToPreferredWiFi() {
  // Введіть дані вашої перевіреної мережі const char* preferredSSID = "vallyk97"; const char* preferredPassword = "48973592";

  Serial.println("Спроба підключення до перевіреної мережі Wi-Fi...");

  WiFi.begin(preferredSSID, preferredPassword);

  unsigned long startAttemptTime = millis(); // Початок спроби підключення while (WiFi.status() != WL_CONNECTED) { delay(500); Serial.print(".");
    if (millis() - startAttemptTime > 10000) { // Якщо пройшло більше 10
секунд, вважаємо це невдалою спробою
      Serial.println("Не вдалося підключитися до перевіреної мережі Wi-
Fi.");
      return; // Виходимо з функції, оскільки не вдалося підключитися до
переданої мережі
    }
  }

  // Якщо підключено успішно, виводимо IP-адресу та продовжуємо виконання програми
```

```
      Serial.println("");
      Serial.println("Успішно підключено до перевіреної мережі Wi-Fi!");
      Serial.println("Ваш IP-адреса " + WiFi.localIP().toString());
}

    void connectWiFi() {
      // Спочатку спробуємо підключитися до заданої мережі connectToPreferredWiFi();

      // Якщо не вдалося підключитися до заданої мережі, виконаємо процес сканування та підключення до іншої мережі   if (WiFi.status() != WL_CONNECTED) {
        Serial.println("Не вдалося підключитися до перевіреної мережі Wi-Fi.
    Запускаємо процес сканування мереж...");
        while (true) {       boolean wifiFound = false;
          int i, n;

          WiFi.mode(WIFI_STA);                    WiFi.disconnect(); delay(100);
          Serial.println("Проведення сканування мереж Wi-Fi..."); int nbVisibleNetworks = WiFi.scanNetworks(); Serial.println(F("Сканування завершено!"));     if (nbVisibleNetworks == 0) {
            Serial.println("Мережі Wi-Fi не знайдено!"); continue;
          }
          Serial.print(nbVisibleNetworks);
          Serial.println(" мереж знайдено");

          // Виводимо список доступних мереж        for (i = 0; i < nbVisibleNetworks; ++i) {
            Serial.println(WiFi.SSID(i));
          }
```

```
            // Введення імені мережі та пароля через Serial Monitor
            Serial.println("Введіть SSID мережі, до якої ви хочете
підключитися:");         while (!Serial.available()) {} // Очікування
вводу
            String ssidInput = Serial.readStringUntil('\n');
            Serial.println("Введіть пароль:");
            while (!Serial.available()) {} // Очікування вводу
            String passwordInput = Serial.readStringUntil('\n');

            // Перевірка, чи знайдено введену мережу в списку доступних
for (i = 0; i < nbVisibleNetworks; ++i) {          if (WiFi.SSID(i) ==
ssidInput) {            wifiFound = true;            break;
              }
            }
            if (!wifiFound) {
              Serial.println("Введена мережа не знайдена!");
              continue;
            }

            // Підключення до введеної мережі
            Serial.print("Підключення до ");
            Serial.println(ssidInput);
            WiFi.begin(ssidInput.c_str(),      passwordInput.c_str());
unsigned long startAttemptTime = millis(); // Початок спроби
     підключення            while  (WiFi.status()  !=  WL_CONNECTED)  {
delay(500);          Serial.print(".");
              if (millis() - startAttemptTime > 10000) { // Якщо пройшло
більше 10
     секунд, вважаємо це невдалою спробою
                Serial.println("Час       очікування       підключення
вичерпано!");            break; // Виходимо з циклу
              }
            }
```

```
          if (WiFi.status() == WL_CONNECTED) { // Якщо підключено успішно,
     виходимо з циклу
            Serial.println("");
            Serial.println("Успішно підключено до мережі Wi-Fi!");
Serial.println("Ваш IP-адреса " + WiFi.localIP().toString());
break;
          } else { // Якщо підключення не вдалося
            Serial.println("Помилка підключення, повторна спроба...");
          }
        }
      }
     }

     void checkWiFiConnection() {  if (WiFi.status() != WL_CONNECTED) {
        Serial.println("Втрачено з'єднання з Wi-Fi. Повторна спроба підключення..."); connectWiFi(); // Повторне підключення до Wi-Fi
      }
     }

     void setupTelegram() {
      Serial.println("Підключення Telegram bot...");
secured_client.setTrustAnchors(&cert); // Додати кореневий сертифікат для api.telegram.org

      configTime(0, 0, "pool.ntp.org"); // Отримати час UTC через NTP
time_t now = time(nullptr);  while (now < 24 * 3600) {  delay(100);
now = time(nullptr);
      }

      Serial.println("Telegram bot підключено!");
```

```
}

void handleNewMessages(int numNewMessages) {
  Serial.print("handleNewMessages ");
  Serial.println(numNewMessages);

  for (int i = 0; i < numNewMessages; i++) {
    String text = bot.messages[i].text;

    if (text == "/temp") {
      String message = "Температура: " + String(T) + "°C\n";
bot.sendMessage(bot.messages[i].chat_id, message, "");
    } else if (text == "/pres") {
      String message = "Тиск: " + String(P) + " мм. рт. ст.\n";
bot.sendMessage(bot.messages[i].chat_id, message, "");
    } else if (text == "/lux") {
      String message = "Яскравість освітлення: " + String(L) + "
люкс\n";       bot.sendMessage(bot.messages[i].chat_id, message, "");
    } else if (text == "/all") {
      String message = "Температура: " + String(T) + "°C\n";
message += "Тиск: " + String(P) + " мм. рт. ст.\n";       message +=
"Яскравість освітлення: " + String(L) + " люкс\n";       message +=
"Рівень напруги: " + String(vcc) + " V\n";       message += "Рівень
сигналу:      "      +      String(WiFi.RSSI())      +      "dBm\n";
bot.sendMessage(bot.messages[i].chat_id, message, "Markdown");
    } else if (text == "/help") {
      String welcome = "ESP8266 WiFi Telegram !\n";
      welcome   +=   "Для   отримання   даних   оберіть   потрібну
команду:\n\n";       welcome += "/temp : Температура\n";       welcome
+= "/pres : Тиск\n";       welcome += "/lux : Яскравість освітлення\n";
welcome       +=       "/all       :       Загальні       значення\n";
bot.sendMessage(bot.messages[i].chat_id, welcome, "Markdown");
    }
  }
}
```

```
    void measure() {   Wire.begin();   vcc = ((float)ESP.getVcc()) /
1000.0;   rssi = WiFi.RSSI();   T = bmp.readTemperature();
      P = bmp.readPressure();
      P = P / 133.322;
      Serial.print("T= ");
      Serial.print(T, 2);
      Serial.print(" C; P= ");
      Serial.print(P , 2);
      Serial.print(" mmHg; VCC= ");
      Serial.print(vcc, 3);
      Serial.print(" V, RSSI= ");
      Serial.print(WiFi.RSSI());
      Serial.println(" dBm ");
    }

    void luxt() {    Wire.begin();    lightMeter.begin();    L =
lightMeter.readLightLevel();
      Serial.print("L= ");
      Serial.print(L, 2);
      Serial.print(" Lx ");
    }

    void narodmonSend() {    unsigned long currentTime = millis();
// Перевіряємо, чи минуло вже narodmonSendInterval мілісекунд    if
(currentTime - lastNarodmonSendTime >= narodmonSendInterval) {    if
(!client.connect(host, httpPort)) {
          Serial.println("Connection    to    narodmon    failed");
return;
        }
        Serial.println("Sending    data    on    narodmon    ...");
client.print("#");                  client.print(WiFi.macAddress());
client.print("#");    client.print("ESP8266+BMP280");
        client.println();                  client.print("#temp#");
client.println(T);    client.print("#pres#");    client.println(P);
```

```
client.print("#lx#");                                client.println(L);
client.print("#rssi#");                             client.println(rssi);
client.print("#vcc#");                              client.println(vcc);
client.println("##");     delay(10);
         Serial.print("Requesting: ");     while (client.available())
{
           String line = client.readStringUntil('\r');
           Serial.println(line);
         }     client.stop();     Serial.println("narodmon ok!");

         //      Оновлюємо      час      останнього      надсилання
lastNarodmonSendTime = currentTime;
       }
     }

     void dweetSend() {
       if        (!client.connect(dweet,        dweetPort))        {
Serial.println("connection dweet failed");     return;
       }
       client.print(String("GET
/dweet/for/ESP8266_BMP280?Temperature=")  +
     String(T, 2)
                    + "&Pressure=" + String(P, 2)
                    + "&Lx=" + String(L, 2)
                    + "&RSSI=" + String(rssi)
                    + "&VCC=" + String(vcc, 3)
                    + "&Uptime=" + String(millis() / 1000)
                    + " HTTP/1.1\r\n" +
                    "Host:     "     +     dweet     +     "\r\n"     +
"Connection: close\r\n\r\n");   delay(10);
       while (client.available()) {
         String line = client.readStringUntil('\r');
         Serial.print(line);
       }
       client.stop();
```

```
      Serial.println("dweet ok!");
    }

    void setup() { Serial.begin(115200); bmp.begin(0x76);
connectWiFi(); setupTelegram();
      Serial.print("MAC адреса: ");
Serial.println(WiFi.macAddress()); server.on("/", HTTP_GET,
handleWebpage); server.on("/data", HTTP_GET, handleData);
server.begin();
    }

    void loop() { if (millis() > lasttime + interval) { lasttime
= millis(); measure(); luxt();
        checkWiFiConnection(); narodmonSend(); dweetSend();
int numNewMessages = bot.getUpdates(bot.last_message_received + 1);
while (numNewMessages) {
          Serial.println("got response");
handleNewMessages(numNewMessages); numNewMessages =
bot.getUpdates(bot.last_message_received + 1);
        } lasttime = millis();
      } server.handleClient();}
```

Завідувачу кафедри КІІС
д-р. філософії, доц. Ользі ПАВЛОВІЙ

Валентина КОСТЬОЛКА

ПІБ здобувача вищої освіти

ФІТ, 4 курсу, групи КІ2-21-4

ЗАЯВА

З правилами чинного Положення «Про систему забезпечення академічної доброчесності у Хмельницькому національному університеті» від 01.07.2022, згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений (а). Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений(а) та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Strike-Plagiarism та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

19.06 2025 року

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Дипломник: Костьолко Валентин Русланович

Тема: Кіберфізична система прогнозування опадів на основі ESP8266

Спеціальність: 123 «Комп'ютерна інженерія»

Обсяг кваліфікаційної роботи:

Кількість листів креслень ___3____ Кількість сторінок записки ___83___

1. Короткий зміст роботи та прийнятих рішень: Метою кваліфікаційної роботи є створення системи віддаленого моніторингу якості води у водосховищах на основі ESP32

2. Висновок про відповідність роботи дипломному завданню: Робота повністю відповідає поставленому завданню.

3. Характеристика виконання кожного розділу, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У першому розділі проведено аналіз архітектурних та функціональних засад кіберфізичних систем, призначених для моніторингу метеорологічних параметрів. Встановлено, що впровадження автоматизованих засобів вимірювання температури, вологості, атмосферного тиску та опадів є критично важливим для побудови адаптивних систем підтримки прийняття рішень у сферах агропромисловості, транспорту та енергетики. Проаналізовано приклади застосування метеорологічного моніторингу у практичних задачах, включаючи управління іригаційними системами, адаптивну логістику та прогнозування вироблення енергії з відновлюваних джерел. Особливу увагу приділено ролі метеоданих у забезпеченні стабільності енергетичних систем в умовах децентралізованої генерації.

У другому розділі здійснено відбір компонентів метеосистеми: ESP8266 (NodeMCU), сенсори DHT22, BME280, FC-37, дисплеї SSD1306/ILI9341. Обґрунтовано архітектуру системи, типи інтерфейсів, режими живлення (AMS1117, TP4056, Li-Ion 18650), методи інтеграції з Arduino IDE. Надано технічні характеристики, обґрунтовано точність і сумісність.

У третьому розділі реалізовано логіку функціонування системи, зокрема алгоритм зчитування, фільтрації та передавання даних через MQTT/HTTP. Використано бібліотеки Adafruit_BME280.h, ESP8266WiFi.h, ThingSpeak.h. Проведено тестування авторизації, підключення до Wi-Fi, обробки JSON-даних. Система функціонує як автономний вузол IoT із підтримкою віддаленого моніторингу.4. Позитивні сторони роботи: високий ступінь технічної деталізації. Робота містить обгрунтований вибір апаратних компонентів (ESP8266, BME280, FC-37) з наведенням точних параметрів, протоколів та стандартів

5. Негативні сторони роботи: робота не містить емпіричної оцінки точності вимірювань у порівнянні з еталонними пристроями або комерційними аналогами.

6. Оцінка графічного оформлення та пояснювальної записки роботи: Пояснювальна записка оформлена коректно, згідно діючих стандартів оформлення документації.

7. Відгук про роботу в цілому: Робота виконана на належному науково-технічному рівні.

8. Інші зауваження: ______________________________

9. Оцінка дипломної роботи: задовільно

Рецензент (прізвище, ім’я, по батькові, посада, місце роботи)

Федорак Михайло Васильович др філософії, ст. викладач, к.ф. кібербезпеки

“19” 06 2025 р. ______________(підпис)

РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Кіберфізична система прогнозування опадів на основі ESP8266

Автор: Валентин КОСТЬОЛКО

Спеціальність: 123– Комп'ютерна інженерія

Освітня програма: освітньо-професійна

Науковий керівник: Дмитро ДЕНИСЮК, старший викладач

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) запозичення розміщено виключно у розділах, присвячених аналізу існуючих технічних рішень, аналогів та прототипів, які мають оглядовий характер і не містять оригінальних результатів дослідження автора;
2) усі випадки текстових збігів є фрагментарними та супроводжуються належним чином оформленими бібліографічними посиланнями на джерела відповідно до вимог академічної доброчесності;
3) частина виявлених збігів стосується загальновживаних технічних або термінологічних формулювань, що підтверджується наявністю ідентичних фрагментів у десятках різних джерел, що виключає їх оригінальність як авторського тексту;
4) також було виявлено збіги, що стосуються специфічних позначень у формулах, які включають поєднання латинських символів з українськими скороченнями індексів.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості StrikePlagiarism, складає 1.1% та системою Anti-Plagiarism складає 0.0%, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ Дмитро ДЕНИСЮК

Гарант ОП _____ Андрій НІЧЕПОРУК

Завідувач кафедри КІІС _____ Ольга ПАВЛОВА

Thu Jun 19 07:26:52 EEST 2025, Медзатий Дмитро Миколайович, Хмельницький національний університет, ХНУ

Anti-Plagiarism (UA) v-15.281 Educational

The maximum coincidence with one document 0.0%

Dictionaries check: en_US, ru_RU, ua_UA. **Errors in the documents: 13%**

ID: 246843 Title: БКР Кіберфізична система прогнозування опадів на основі ESP8266 Added in a DB: 2025-06-19 Authors: Валентин КОСТЬОЛКО Heads: Дмитро ДЕНИСЮК Consultants: Opponents:	Document		Sum coincidence on the DB	
	Symbols	Lexemes	Symbols	Lexemes
	123491	921	487 (0%)	6 (1%)

Plagiarism sources

ID	Description	Plagiarism presence in the document	
		Symbols	Lexemes

Протокол аналізу звіту подібності експертом

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Валентин КОСТЬОЛКО

Співавтор:

Назва: Костьолко_Кіберфізична система прогнозування опадів на основі ESP8266

Експерт:

Підрозділ: Кафедра комп'ютерної інженерії та інформаційних систем

Коефіцієнт подібності 1:1.1%

Коефіцієнт подібності 2:0.2%

Мікропробіли: 61

Заміна букв: 1

Інтервали: 0

Білі знаки: 2

Дата створення звіту: 2025-06-19 06:34:53.0

Після аналізу Звіту подібності констатую наступне:

☒ **Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.**

☐ **Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.**

☐ **Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедурам. Таким чином робота не приймається.**

Обгрунтування:

2025-06-19 Доцент Андрій Нічепорук

Дата експерт