Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут деревообробних та
комп'ютерних технологій і дизайну
(повне найменування інституту, назва факультету(відділення))

Кафедра інформаційних систем і комп'ютерного моделювання
(повна назва кафедри (предметної циклової комісії))

Пояснювальна записка

до дипломної роботи

перший (бакалаврський)

(рівень вищої освіти)

на тему: «Розроблення цифрової системи метеорологічних спостережень»

Виконав студент 4 курсу, групи ІСТ-41
спеціальності: 126
„Інформаційні системи та технології"
(шифр і назва напрямку підготовки спеціальності)

Артемук Андрій Сергійович
(прізвище, ініціали)

Керівник: проф.Шабатура Ю.В.
(прізвище, ініціали)

Рецензент: Кроленний І.М.
(прізвище, ініціали)

Львів-2023

Національний лісотехнічний університет України

(повне найменування вищого навчального закладу)

ННІ деревооброблювальних та комп'ютерних технологій і дизайну

Кафедра Інформаційних систем і комп'ютерного моделювання

Рівень вищої освіти перший (бакаравський)

Спеціальність 126 *„Інформаційні системи та технології"*

ЗАТВЕРДЖУЮ:

В.о завідувача кафедри ІСКМ

_____Сторожук О.Л.

„21" 11 2022.

ЗАВДАННЯ

НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Артемук Андрій Сергійович

(прізвище, ім'я, по батькові)

1.Тема магістерської роботи: Розроблення цифрової системи метеорологічних спостережень.

керівник роботи проф. Шабатура Ю.В.,

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

затверджені наказом вищого навчального закладу від "21" 11 2022 року, №С-521

2.Термін подання студентом проекту(роботи) 10 червня 2023р

3. Вихідні дані до проекту (роботи) Розробити програмне забезпечення (прошивку) для метеорологічної станції. Програмне забезпечення має надавати можливість підключення до мережі Ethernet, зчитувати дані вимірювань із сенсорів та надавати результати вимірювань за HTTP запитом.

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

Стан проблемної області

Інформаційне забезпечення

Програмне забезпечення

Висновки

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

Підготовка матеріалу до доповіді .

6. Консультанти розділів проекту (роботи)

7. Дата видачі завдання 23.11.2022р.

КАЛЕНДАРНИЙ ПЛАН

№, з/п	Етапи бакалаврської роботи	Термін виконання етапів роботи	Примітка
1.	Огляд літератури згідно досліджуваної теми. Збір необхідних матеріалів.	23.11-20.12	Викон.
2.	Постановка задачі і її формалізація	20.12-13.01	Викон.
3.	Виконання вхідного етапу технології	13.01-14.04	Викон.
4.	Реалізація головних класів проекту	14.04-20.05	Викон.
5.	Виконання етапу відлагодження проєкту	20.05.-25.05.	Викон.
6.	Виконання етапу впровадження та випуску бета-версії.	25.05.-02.06.	Викон.
7.	Оформлення записки до дипломного проекту.	02.06.-10.06.	Викон.

Студент _____ Артемук Андрій Сергійович
(підпис) (прізвище та ініціали)

Керівник роботи _____ проф.Шабатура Ю.В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Дипломна робота містить 37 сторінок пояснювальної записки, 29 рисунків, 1 додаток, 15 джерел.

Дипломна робота присвячена розробці серверного програмного забезпечення для пристрою, що вимірює параметри зовнішнього оточуючого середовища із можливістю розміщення в просторі, а також може використовуватись як компонент розумного дому. У роботі обґрунтовано актуальність даного проєкту та реалізоване програмне забезпечення засобами середовища Arduino IDE, а також мовою програмування C. Програмний інтерфейс для підключення інтуїтивно зрозумілий не потребує додаткових програмних та апаратних компонентів.

Ключові слова: Arduino, ESP8266, прошивка, програмне забезпечення.

ABSTRACT

Thesis contains 37 pages of explanatory note, 29 drawings, 1 appendix, 15 sources.

The thesis is devoted to the development of server software for a device that measures the parameters of the external environment with the possibility of placement in space, and can also be used as a component of a smart home. The work substantiates the relevance of this project and the implemented software using the Arduino IDE environment, as well as the C programming language. The software interface for connection is intuitive and does not require additional software and hardware components.

Keywords: Arduino, ESP8266, Firmware, software.

ТЕХНІЧНЕ ЗАВДАННЯ

Для пристрою створеного на основі мікроконтролера і сенсорів для вимірювання температури та тиску створити програмне забезпечення (прошивку, firmware). Мікроконтролер що використовується у пристрої — ESP8266 фірми Espressif Systems. Сенсора для вимірювань BMP-280 фірми Bosch Sensortec GmbH. Програмне забезпечення має надавати можливість підключення до мережі Ethernet, зчитувати дані вимірювань із сенсорів та надавати результати вимірювань за HTTP запитом.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

MCU – мікроконтролер (Microcontroller Unit).

UART – універсальний асинхронний приймач-передавач (Universal Asynchronous Receiver-Transmitter).

I2C – послідовна шина передачі даних між інтегральними схемами (Inter-Integrated Circuit).

SPI – синхронний послідовний периферійний інтерфейс (Serial Peripheral Interface).

HTTP – протокол передачі гіпертексту (HyperText Transfer Protocol).

IDE – інтегроване середовище розробки (Integrated Development Environment).

HTML – мова розмітки гіпертексту (HyperText Markup Language).

API – інтерфейс програмування застосунків (Application Programming Interface).

ВСТУП

Якщо ви цікавитеся метеорологією, ви, напевно, розглядаєте можливість придбання одного з численних наявних метеорологічних приладів або метеостанції, чи не так? На ринку є багато моделей, але деякі з них є більш повними, ніж інші, залежно від їх ціни. Найбільш дорогі пристрої можуть вимірювати більше кліматичних змінних, що робить їх ідеальними для тих, хто бажає глибше досліджувати клімат у своєму регіоні, тоді як недорогіші моделі задовольнять потреби користувачів, які цікавляться переважно температурою, а можливо, і вологістю.

Залежно від ваших потреб, цікаво дізнатися про різні типи метеорологічних приладів і їх функції. Це допоможе вам зробити більш обізнанений вибір підходящої моделі для вас.

Об’єктом дослідження використання Arduino для створення домашньої метеостанції.

Метою роботи збирання важливих даних, які сприяють розумінню погодних явищ і кліматичних змін в тропосфері

Предметом дослідження є створення гнучкого програмного забезпечення то приладу для вимірювання показників зміни погодних явищ.

Практичне значення роботи полягає у простоті та дишевизні реалізації систем такого типу.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Огляд проблемної області.

Основна мета цих станцій полягає у вивченні нижньої кулі атмосфери, відомої як тропосфера. На таких станціях використовуються спеціальні прилади для дослідження різних погодних умов. Вони дозволяють виміряти температуру повітря та його вологість, встановити швидкість та напрямки вітру, а також зафіксувати показники барометра і атмосферного тиску.

Ці метеорологічні станції збирають важливі дані, які сприяють розумінню погодних явищ і кліматичних змін в тропосфері. Інформація про температуру та вологість повітря дозволяє встановити зв'язок між погодними умовами і кліматичними процесами. Вимірювання швидкості та напрямку вітру надають важливі дані для прогнозування погоди і вивчення впливу вітрових систем.

Записи барометра і показники атмосферного тиску важливі для виявлення змін у погодних умовах і відслідковування атмосферних подій, таких як циклони, антициклони та інші метеорологічні явища. Ці дані є цінними для розробки моделей погоди і прогнозування погодних умов у майбутньому.

Таким чином, метеорологічні станції виконують важливу роль у зборі і аналізі даних про погоду і кліматичні зміни, сприяючи покращенню нашого розуміння атмосферних процесів і допомагаючи прогнозувати погоду в різних регіонах.

Також станції часто використовуюють у лісовій сфері. Лісові метеорологічні станції мають важливе призначення - запобігати лісовим пожежам. Ці станції зазвичай живляться від акумуляторів. Вони збирають різноманітні кліматичні дані, включаючи вологість дерев, ґрунту та температуру на різних рівнях висотності в лісах.

Це забезпечує операторам станцій необхідну інформацію для вивчення кліматичних умов, які можуть сприяти появі лісових пожеж. Зокрема, вологість є важливим показником, оскільки сухе середовище може сприяти швидкому поширенню пожежі. Дані про температуру на різних рівнях висотності

відображають кліматичні умови, які можуть впливати на ризик виникнення пожежі.

Ці метеорологічні станції допомагають забезпечити оперативне спостереження за кліматичними змінами і попередити можливість виникнення лісових пожеж. Вони є важливою складовою системи моніторингу і контролю за станом лісових екосистем, допомагаючи зберегти лісові масиви від знищення та зберегти природний рівновагу.

1.2. Як вибрати метеостанцію

Вибираючи метеостанцію, слід враховувати кілька факторів, таких як параметри погоди, які ви хочете виміряти, і ваш бюджет. Тут ви знайдете відповіді на деякі з найпоширеніших питань про те, як вибрати метеостанцію, і які допоможуть вам у цьому процесі.

Метеостанція є складною одиницею, оскільки вона має багато приладів, які використовуються для вимірювання змінних погоди. Метеостанції бувають різних форм і розмірів, але також мають функції, призначені для різних цілей, таких як вимірювання температури, вологості, дощу та швидкості вітру.

При виборі важливо подумати про призначення метеостанції. Якщо ви просто хочете виміряти температуру або вологість у своїй кімнаті, покупка метеостанції не є хорошим вибором.

1.2.1. Використання в побуті

Домашня метеостанція, такі фактори, як температура, вологість, барометричний тиск, швидкість вітру та інші метеорологічні змінні, вимірюються з мінімальним обладнанням. Отже, ці типи метеорологічних приладів в основному призначені для використання вдома. Їх можна розділити на дві категорії:

а. Базова домашня метеостанція

Ці метеостанції створені спеціально для використання в домашніх умовах. Вони часто мають дисплей, а також невеликий зовнішній датчик. Інші датчики, такі як датчики вітру та температури, повністю залежать від обраної моделі. Якщо ви можете витратити трохи додаткових грошей, ви можете отримати метеостанцію з усіма розширеними функціями.

b. Повна домашня метеостанція

Ці метеостанції ідеально підходять для любителів погоди, які, ймовірно, будуть в курсі останніх прогнозів. Ці метеостанції оснащені всіма датчиками метеостанції, а також додатковим датчиком дощу. Для людей, які більшу частину часу проводять на вулиці, цей тип метеостанцій надзвичайно корисний

1.2.2. Професійне використання

Ці типи метеостанцій часто розробляються для професіоналів, яким регулярно потрібні дані про погоду та прогнози.

Однією з унікальних особливостей професійних метеостанцій є те, що вони мають дивовижний дизайн.

Більшість професійних метеостанцій надають дані в режимі реального часу, оскільки саме вони відповідають метеорологічним стандартам. В основному професійні метеостанції використовуються з сільськогосподарських причин і включають такі датчики, як датчики УФ-випромінювання, датчики температури ґрунту та датчики температури води.

Вони найбільш підходять для освітніх, комерційних, промислових та військових цілей моніторингу.

Важливо враховувати місце, де буде використовуватися станція, перш ніж купувати її. Є райони, де погода може швидко змінюватися. Місце може мати більшу швидкість вітру, ніж у середньому, з частішими змінами.

Якщо у вас вже є метеостанція, яка оновлюється повільніше, то надана вам інформація не буде відповідати поточним погодним умовам. Таким чином, важливо, щоб вибрана вами метеостанція тримала вас в курсі цих змін у вашому регіоні.

Важливо враховувати місце, де буде використовуватися станція, перш ніж купувати її. Є райони, де погода може швидко змінюватися. Місце може мати більшу швидкість вітру, ніж у середньому, з частішими змінами.

Якщо у вас вже є метеостанція, яка оновлюється повільніше, то надана вам інформація не буде відповідати поточним погодним умовам. Таким чином, важливо, щоб вибрана вами метеостанція тримала вас в курсі цих змін у вашому регіоні.

Ще одним фактором, який слід враховувати, є стан клімату в даній місцевості. Деякі метеостанції не є водонепроникними, тому вони не рекомендуються для місць з морозом та обмерзанням. Ви можете вибрати високоякісні метеостанції, оскільки вони побудовані для роботи в будь-яких погодних умовах.

1.3. Спосіб приєднання датчика до медіастанції

Залежно від способу підключення, існують два типи метеостанцій - провідні і бездротові. Який з них вибрати найкраще? Зрозуміло, що бездротові метеостанції є найпопулярнішим варіантом. Це має свої причини. По-перше, вони не потребують кріплення поруч з вікном. Ви можете розмістити таку метеостанцію будь-де, і навіть перенести її з однієї кімнати в іншу. Датчик же залишається на місці, де його встановили. По-друге, бездротові метеостанції дозволяють підключати безліч датчиків, що дуже зручно. Датчики можуть бути розміщені в різних місцях, і кожен з них передаватиме дані на метеостанцію. Вся ця інформація буде відображатися на дисплеї пристрою. Єдине, на що треба звернути увагу, це радіус дії датчиків. Тобто, на якій відстані від метеостанції

можна встановлювати датчики. Зазвичай, цей параметр варіюється від 30 до 100 метрів. Тому важливо обрати модель з оптимальним радіусом дії.

Дротові метеостанції розміщують не далеко вд датчика. Радіус дії залежить від довжиною проводу підключення. В такому разі багато датчиків підключити не вдасться. Проте такі моделі відзначаються дишевизною, в чому основна їх перевага.

1.4. Живлення

В залежності від джерела живлення, метеостанції можуть бути розділені на два типи: автономні і мережеві. Кожен з них має свої особливості. Автономні метеостанції працюють від батарейок або акумуляторів. Головна перевага таких станцій полягає в їх незалежності від електромережі. Ви можете встановлювати їх де завгодно (в рамках радіусу дії датчиків) і, при необхідності, переносити з одного місця в інше. Вони не використовують електроенергію, тому вам не потрібно платити за неї. Однак, оскільки автономні метеостанції працюють на батарейках, вам постійно доведеться їх замінювати. Це є головним недоліком таких моделей. Особливо часто заміну батарейок (або акумулятора) потрібно проводити взимку. Оскільки вони працюють тривалий час в теплих приміщеннях, в датчиках, розташованих зовні, заряду батарейок вистачає максимум на місяць.

Що стосується мережевих моделей, то, як нескладно здогадатися, вони працюють від електромережі. Вам не потрібно замінювати батарейки або акумулятори, що, безумовно, зручно. Однак, оскільки мережеві метеостанції пов'язані з розеткою, ви не зможете встановити їх в будь-якому місці. Крім того, вони повинні бути розміщені в непрямій близькості до датчиків (в рамках радіусу дії).

Отже, яку модель краще придбати? Найкраще враховувати і тип живлення, і спосіб підключення датчиків до метеостан.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Arduino IDE

Розробка електроніки тепер простіше завдяки програмному забезпеченню arduino (IDE) і платам arduino (апаратне забезпечення). Цей набір допомагає будувати цифрові та інтерактивні пристрої за допомогою інших компонентів. Програмне забезпечення arduino (IDE) - це програмне забезпечення з відкритим вихідним кодом, яке використовується для програмування плат Arduino, і є інтегрованим середовищем розробки, розробленим arduino.cc. Дозволяє писати і завантажувати код на плати arduino. І він складається з безлічі бібліотек і набору прикладів міні-проектів. Програмне забезпечення arduino (IDE) сумісне з різними операційними системами (Windows, Linux, Mac OS X) та підтримує мови програмування (C/C++).

Програмне забезпечення Arduino просте у використанні для початківців або досвідчених користувачів. Він використовується для початку роботи з програмуванням електроніки та робототехніки, а також для створення інтерактивних прототипів. Таким чином, програмне забезпечення Arduino - це інструмент для розробки нових речей. і створювати нові електронні проекти, будь-ким (дітьми, любителями, інженерами, програмістами, ... тощо). Програмний інтерфейс Arduino продемонстровано на рис 2.1.

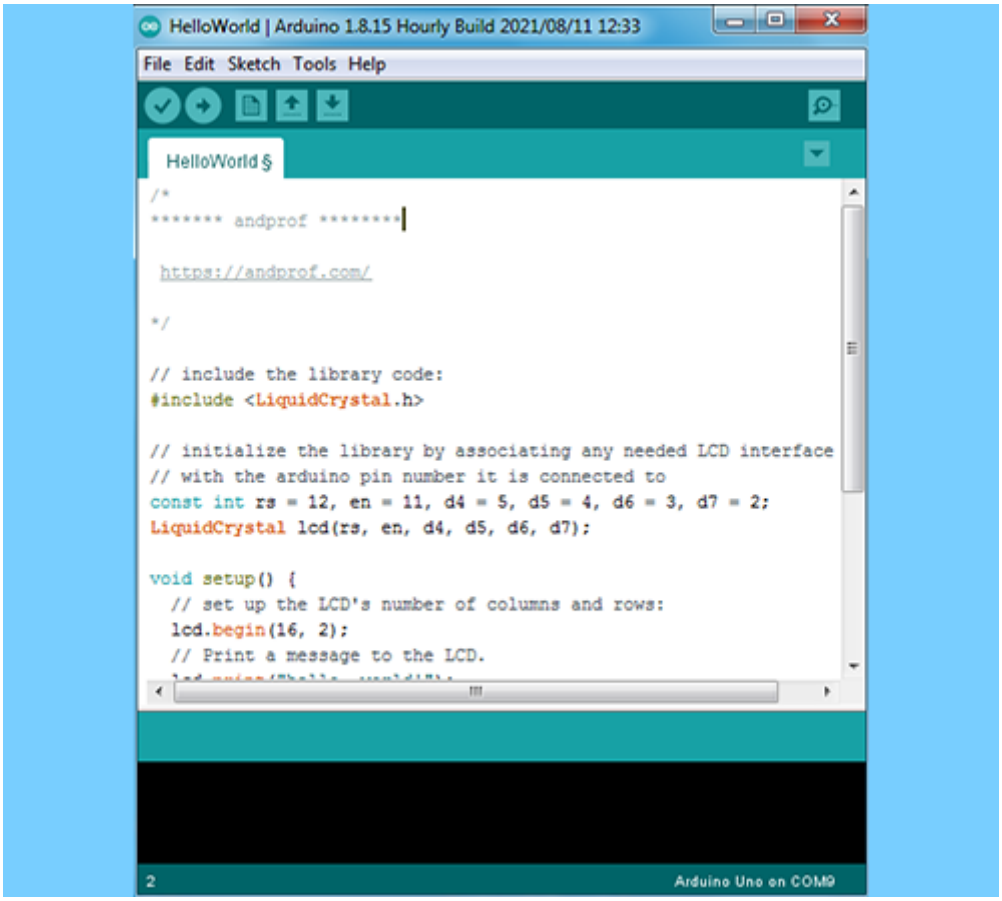

Рис. 2.1. Програмний інтерфейс Arduino

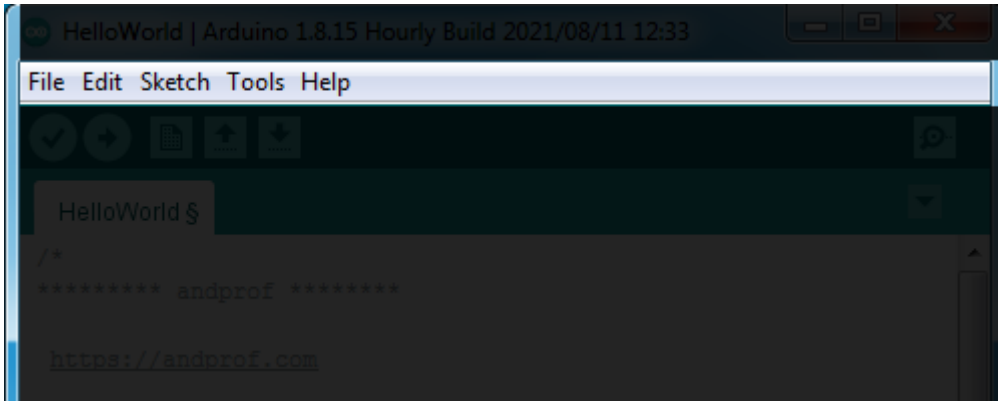

Рис. 2.2. Меню

Меню - це головне меню середовища, і це 5 пунктів для початку роботи (Файл, Редагування, Ескіз, Інструменти, Довідка). Вони використовуються для додавання або зміни коду, який ви пишете. Панель інструментів є найважливішим розділом у програмному забезпеченні Arduino, оскільки вона містить інструменти, які ви будете постійно використовувати під час програмування плати Arduino. Такими інструментами є:

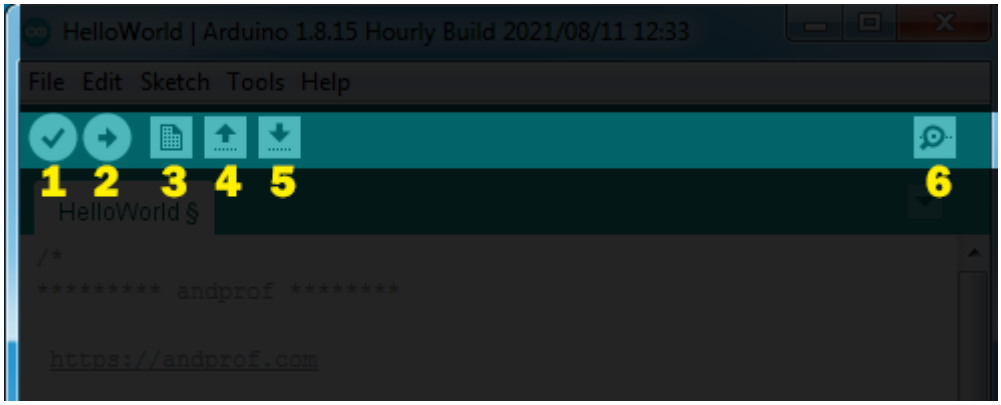

Рис. 2.3. Панель інструментів

1. **Перевірити:** цю кнопку використовуйте, щоб переглянути код або переконатися, що він не містить помилок.
2. **Завантажити:** ця кнопка використовується для завантаження коду на плату arduino.
3. **Новий проєкт:** цю кнопку використовують для створення нового проекту, або ескізу (sketch є файлом коду).
4. **Відкрити:** використовується, коли ви хочете відкрити ескіз з альбому.
5. **Зберегти:** збережіть поточний ескіз у альбомі скетчбуків.
6. **Послідовний монітор:** показує дані, надіслані з arduino.

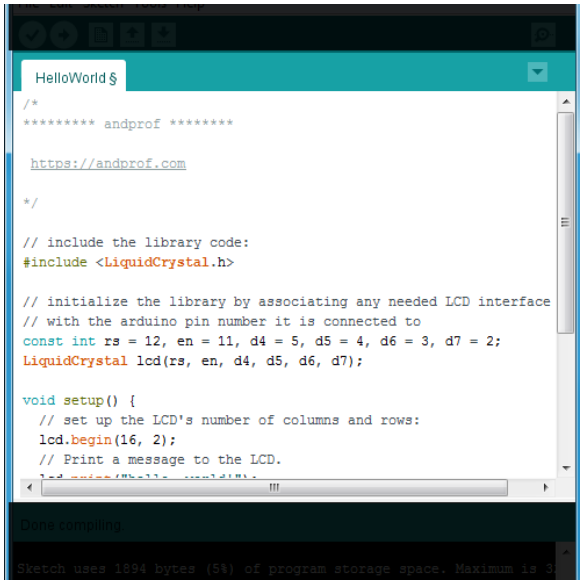

Рис. 2.4. Редактор коду

На Рис. 2.4. та 2.5. можна побачити редактор коду рядок стану.

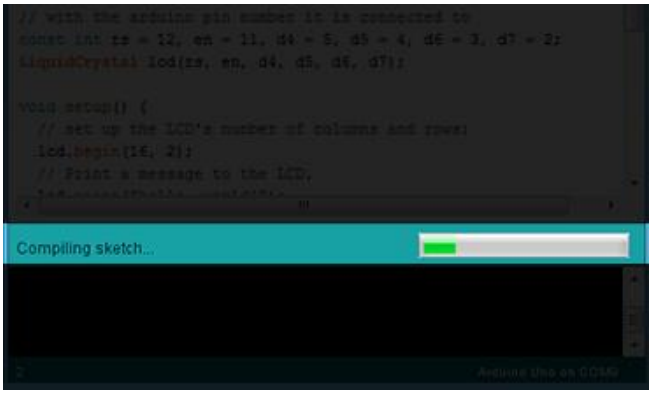

Рис. 2.5. Рядок стану

Рядок стану - це простір, який можна знайти внизу редактора коду, через нього відображається статус завершення операції (компіляція, завантаження, ... тощо). Також є частина, яка відповідає за сповіщення в програмному середовищі. Там можна побачити помилки коду, а також деякі проблеми, з якими ви можете зіткнутися в процесі програмування.

РОЗДІЛ 3. ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Розробка серверної частини.

Для початку продемонструємо структурну діаграму пристрою. Її можна побачити на рис.3.1.

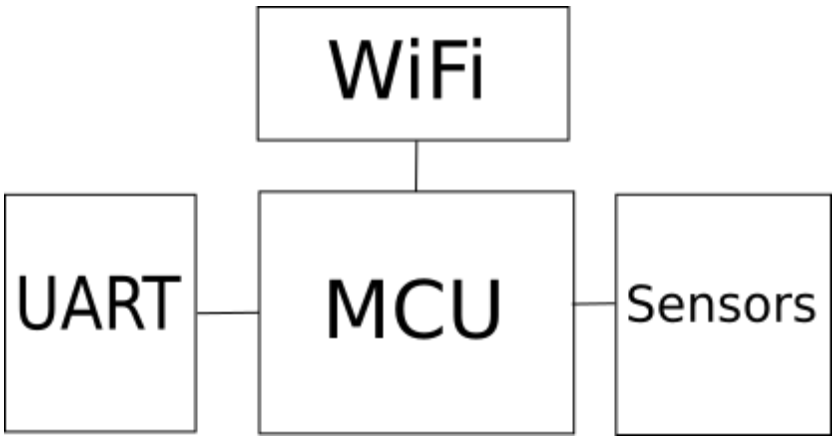

Рис.3.1. Структурна діаграма пристрою

Ядром пристрою є мікроконтролер **MCU** на який покладено функції управління та зв’язку, а також роботу із сенсорами для вимірювання. За допомогою апаратного інтерфейса **UART**, буде відбуватись програмування мікроконтролера, а також виводитись службова інформація для діагностики та відлагодження готового пристрою.

Інтерфейс **UART**, або універсальний асинхронний передавач-приймач — це вузол призначений для організації зв’язку між цифровими пристроями. Перетворює дані для передачі так, щоб була можливість передати їх іншому аналогічному пристрою. Оскільки технологія була створена в середині 1970-х роках минулого сторіччя, методи перетворення добре стандартизовані, і досі широко застосовується у цифровій техніці. Особливо у однокристальних системах, вбудовуваних системах і відповідно мікроконтролерах.

Інтерфейс **WiFi**, це технологія створена для побудови бездротових мереж, за допомогою радіоканалів. Іншими словами сімейство стандартів зв’язку для комунікації у бездротовій місцевій (локальній) зоні. Широко застосовується у

мобільних пристроях таких як смартфони та ноутбуки. А також для технологій інтернету речей. Організація радіоканалів схематично показана на рис. 3.2.

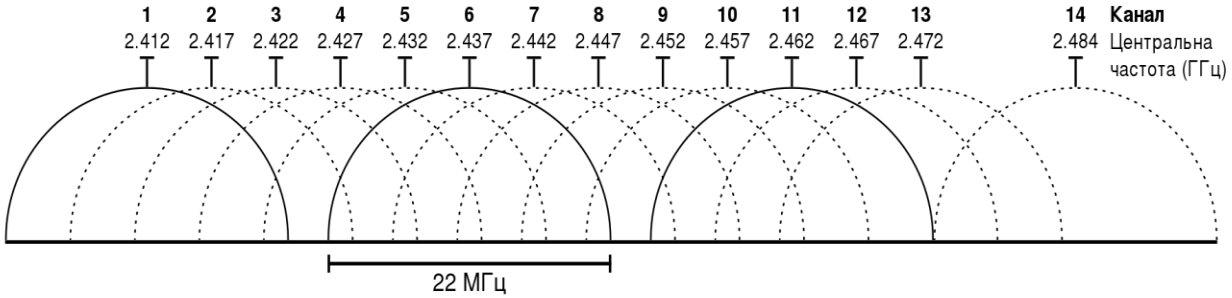

Рис. 3.2. Схема організації радіоканалів технології зв’язку WiFi.

Sensors – цифрові сенсори або датчики. В даному проєкті пристрою використовується датчик для вимірювання атмосферного тиску повітря, і температури. Сучасні сенсори спроєетовані відразу із цифровими інтерфейсами і на виході дають результат вимірювань у одиницях СІ. Більшість цифрових сенсорів мають два типи апаратних інтерфейсів **I2C**, **SPI** або комбіновані.

I2C – послідовна асиметричні шина передачі даних для організації зв’язку всередині цифрових пристроїв. Використовується для комунікації серед низькошвидкісних периферійних компонентів із процесорами та мікроконтролерами. Інтерфейс широко застосовується у пристроях, що передбачають простоту розробки, а також низьку собівартість при відносно вискій швидкодії. Діаграма із прикладом пристрою з інтерфейсом I2C показана на рис. 3

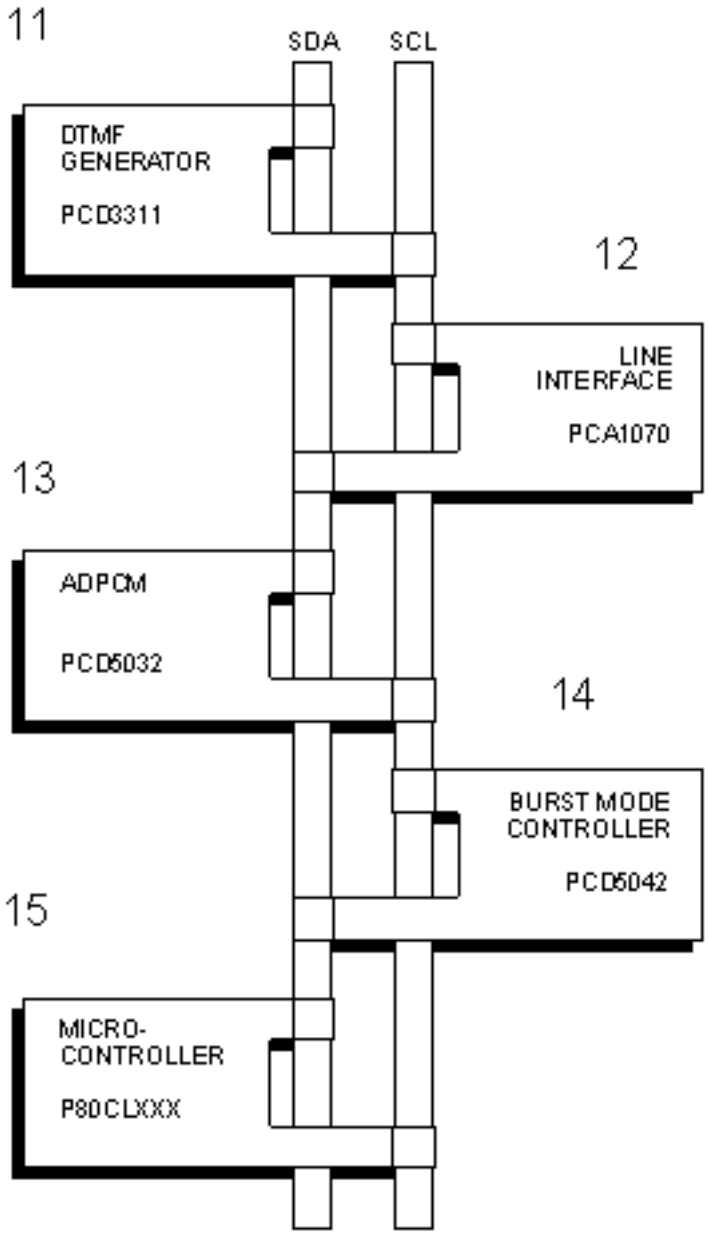

Рис. 3.3. Приклад пристрою з інтерфейсом I2C.

SPI – послідовний периферійний інтерфейс. Послідовний синхронний стандарт передачі данних в режимі одночасного прийому та передачі. Призначення інтерфейсу організація простого і недорогого швидкісного узгодження мікроконтролера та периферії. Навідміну від послідовного інтерфейса **UART**, **SPI** є синхронним інтерфейсом, в якому люба передача або прийом інформації синхронізована спеціальним сигналом, що його генерує головний пристрій — мікроконтролер. Інтерфейс дозволяє з'єднувати між собою кілька пристроїв периферії із мікроконтролером. Існує два способи, або

топології з’єднання “зірка” і “кільце”. Діаграми способів з’єднання показані на рис. 3.4. і рис. 3.5.

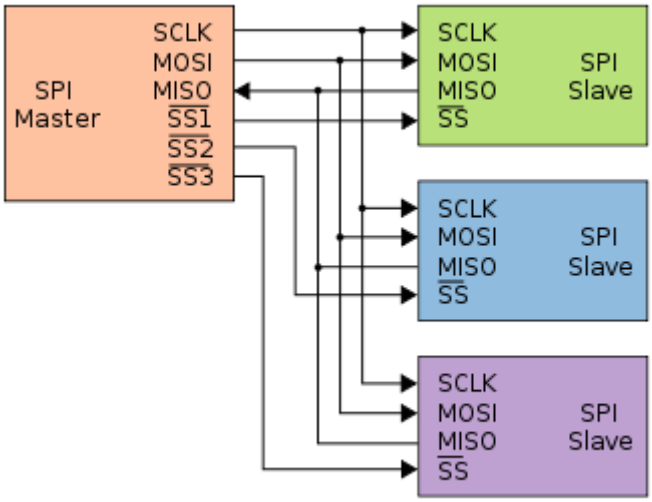

Рис. 3.4. Спосіб з’єднання “зірка”.

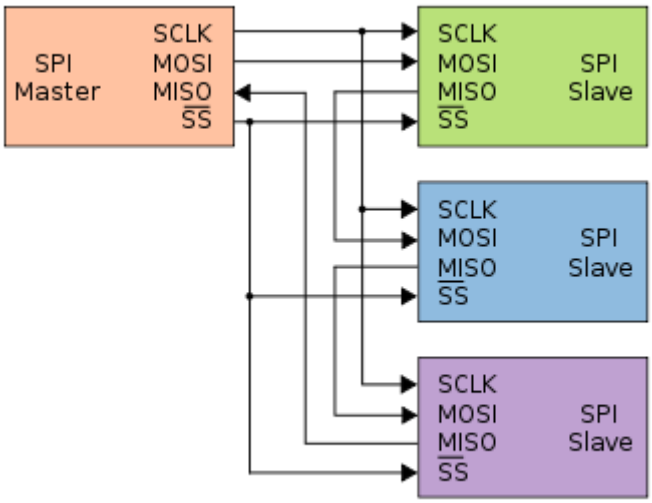

Рис. 3.5. Спосіб з’єднання “кільце”.

3.2. Розробка програмного забезпечення

Для початку роботи налаштуємо інтегроване середовище розробки для роботи із пристроями на базі мікроконтролера ESP8266. Для цього необхідно у налаштуваннях середовища відкрити діалог налаштувань, як це показано на рис. 3.6.

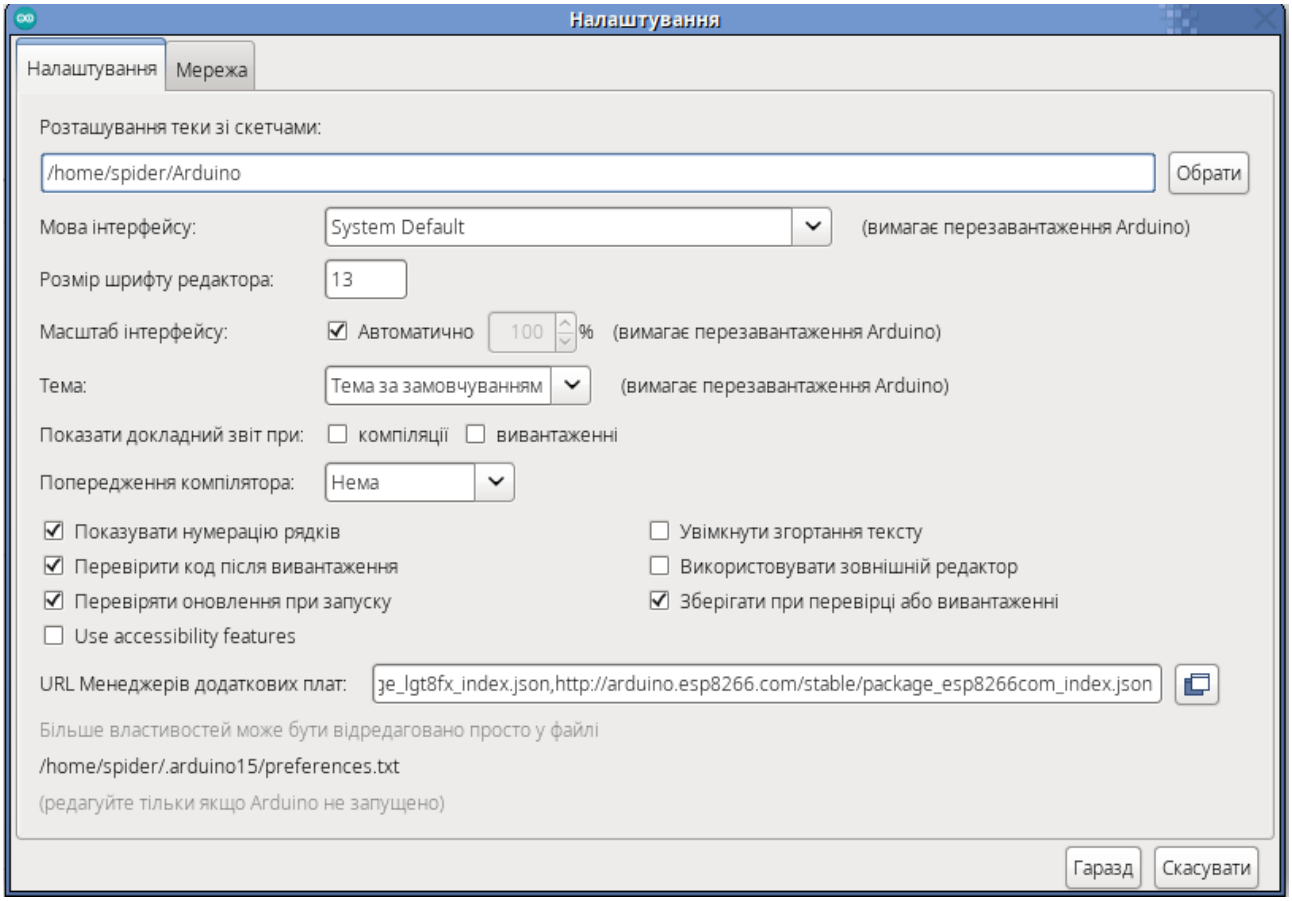

Рис. 3.6. Діалог налаштувань середовища розробки.

Наступним кроком у графу URL Менеджерів додаткових плат вписати адресу менеджера віповідної плати. Для зручності відкриваємо додаткове діалогове вікно, як це показано на рис. 3.7.

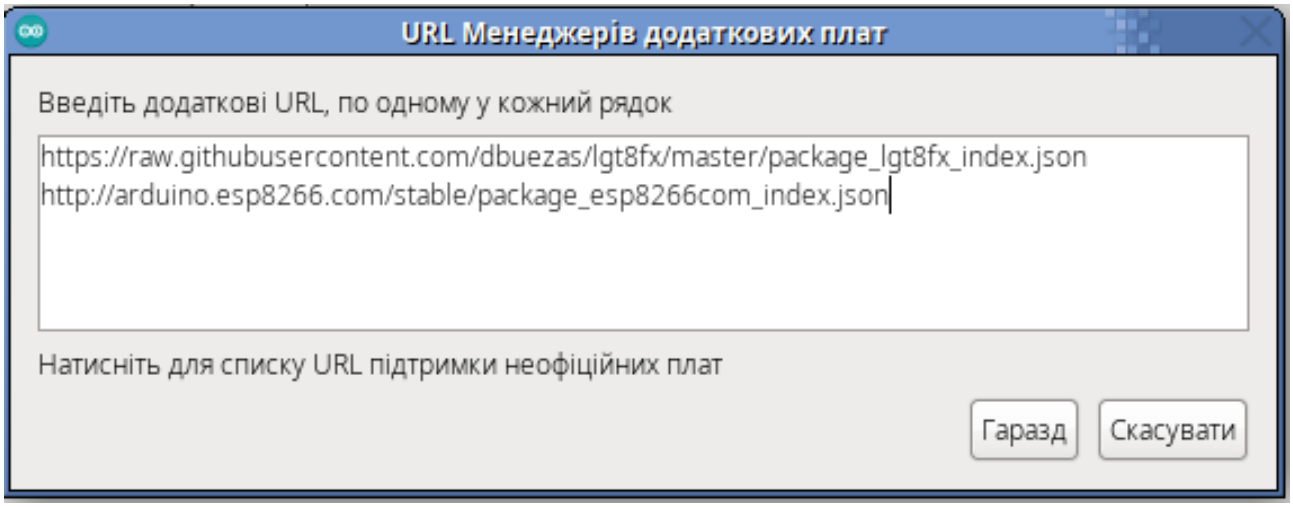

Рис. 3.7. Додаткове діалогове вікно.

Наступним кроком слід у діалоговому вікні менеджера плат вибрати відповідну до технічного завдання плату та встановити програмний інструментарій для неї як це показано на рис. 3.8.

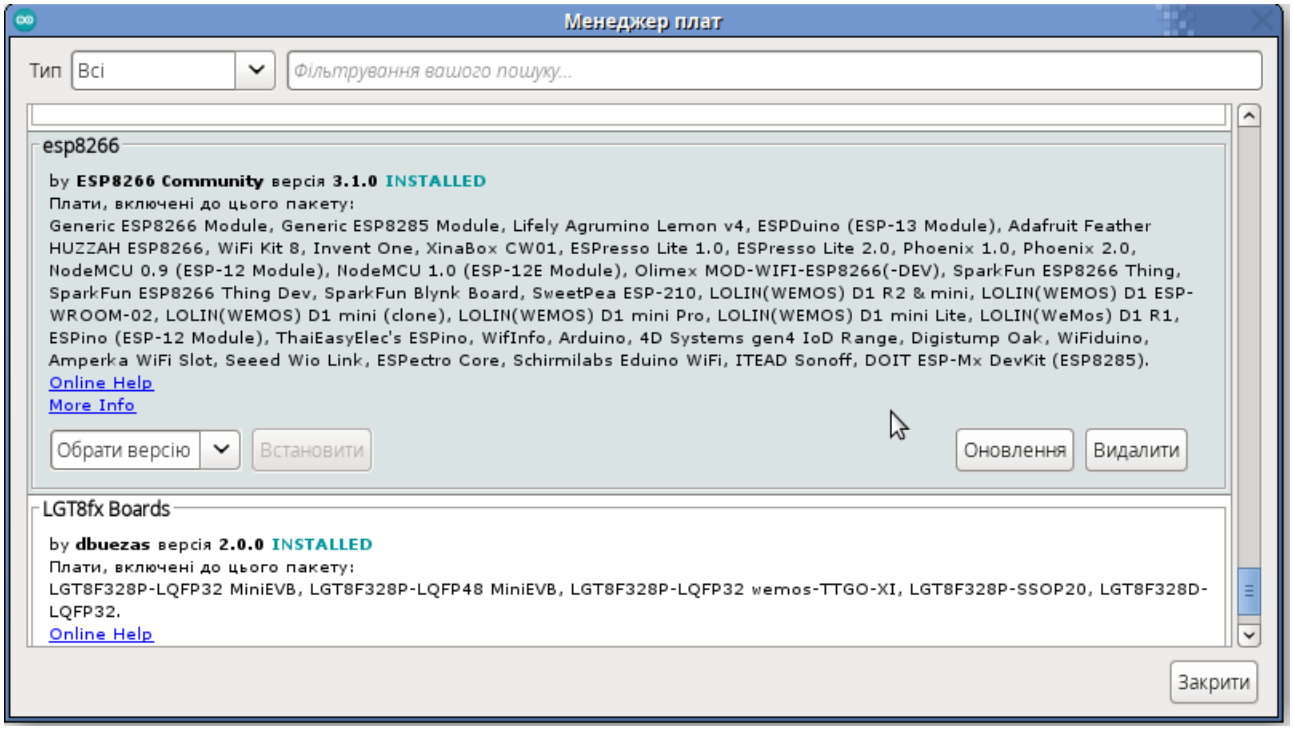

Рис. 3.8. Діалогове вікно менеджера плат

Керуючись технічним завданням обираємо бібліотеки. Сенсори підключені до мікроконтролера за допомогою інтерфейсу I2C. Оскільки мікроконтролер не має вбудованого апаратного інтерфейса, він буде реалізований програмним способом. Інтрефейс SPI зарезервований. За допомогою інтерфейса UART відбуватиметься програмування мікроконтролера а також вивід діагностичної інформації. Штатний обмін даними буде здійснюватись через мережу за допомогою інтерфейса **WiFi**.

Фрагмент коду із підключенням необхідних бібліотек показано на рис. 3.9.

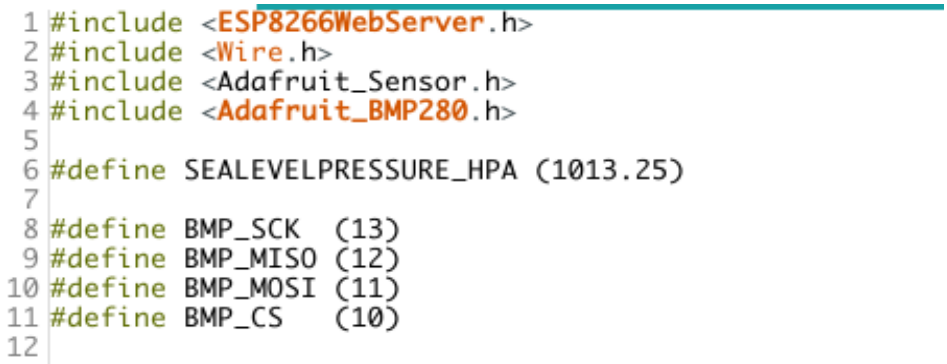

Рис. 3.9. Фрагмент коду із підключенням бібліотек.

Для простоти виводу данних застосуємо веб-сервер що реалізований звсобами бібліотеки ESP8266WebServer.h. Для роботи з інтерфейсом I2C застосуємо бібліотеку Wire.h. Роботу із вимірювальним сенсором побудуємо за допомогою бібліотеки Adafruit_BMP280.h.

Змінна SEALEVELPRESSURE_HPA задає тиск атмосферного повітря на рівні моря, це необхідно для визначення висоти приладу над рівнем моря. Такий метод дозволяє також вимірювати висоту розташування приладу відносно обраної точки. Для цього необхідно внести значення тиску в обраній точці.

Змінними BMP_SCK, BMP_MISO, BMP_MOSI, BMP_CS резервуємо лінії для інтерфейсу SPI.

Результати вимірювань будуть зберігатись у змінних з віповідними назвами.

Temperature – температура

humidity – вологість (зарезевовано)

pressure – тиск атмосферного повітря

altitude – висота над рівнем моря

Тип змінних числа із плаваючою комою. Це необхідно для забезпечення точності результатів вимірювань. Фрагмент коду із описом змінних та ініціалізацією веб-сервера і сенсора показаний на рис. 10

```
12
13 Adafruit_BMP280 bmp; // підключаємось по шині I2C
14
15 float temperature, humidity, pressure, altitude; // задаємо глобальні змінні для вимірювань
16
17 const char* ssid = "*****";  // Вписати назву мережі
18 const char* password = "*****";  // Вписати пароль для підключення
19
20 ESP8266WebServer server(80);
21
```

Рис. 3.10. Фрагмент коду із описом глобальних змінних.

У константах ssid і password зберігатимуться дані бездротової мережі, а саме ідентифікатор мережі ssid і пароль для підключення мережі password. Для передачі даних результатів вимірювань використовується протокол HTTP відтак сервер ініціалізуємо на порту 80, що є стандартним портом.

Наступним кроком задамо режими роботи пристрою за допомогою функції setup (). Фрагмент коду функції показаний на рис. 3.11.

```
22 void setup() {
23   Serial.begin(9600);
24   delay(100);
25
26   bmp.begin(0x76);
27
28   Serial.println("Підключення до ");
29   Serial.println(ssid);
30
31   // запускаємо процес підключення до локальної мережі
32   WiFi.begin(ssid, password);
33
34   //перевіряємо підключення
35   while (WiFi.status() != WL_CONNECTED) {
36   delay(1000);
37   Serial.print(".");
38   }
39   Serial.println("");
40   Serial.println("Мережа підключена..!");
41   Serial.print("Отримано IP адресу: ");  Serial.println(WiFi.localIP());
42
43   server.on("/", OnConnect);
44   server.on("/airtemp", Get_Temperature); //прив'язуємо запит по веб адресі до функції
45   server.on("/airpress", Get_Pressure);
46   server.on("/altitude", Get_Altitude);
47   server.onNotFound(NotFound);
48   server.begin();
49   Serial.println("HTTP сервер увімкнено");
50 }
```

Рис. 3.11. Код функції setup() .

Викликом функції Serial.begin() ініціалізуємо апаратний інтефейс UART. Вхідним значенням задаємо швидкість передачі даних. Цей інтерфейс будемо застосовувати для виводу діагностичної інформації.

При застосуванні кількох сенсорів на одному інтерфейсі слід задати адресу сенсора, що реалізовано викликом функції bmp.begin(). Вхідною змінною функції є адреса пристрою, котра вказана у технічній документації сенсора.

Наступним кроком реалізуємо підключення до бездротової мережі. Викликом функції WiFi.begin (ssid, password) ініціалізуємо інтерфейс бездротової мережі. Вхідною змінною ssid задаємо назву клієнтської мережі а змінною password пароль.

Слідуючий крок це перевірка підключення до бездротової мережі із виводом діагностичної інформації в послідовний порт UART.У циклі while викликаємо функцію WiFi.status () і перевіряємо значення WL_CONNECTED, котре вона повертає. Затримка в 1 секунду потрібна для встановлення підключення. Під час процесу підключення виводитиметься символ “.” . У разі успішного встановлення з’єднання буде виведено повідомлення “Мережа підключена”. Наступним рядком буде виведено отриману IP адресу пристрою.

Слідуючим кроком буде створення вказівників на код, котрий має виконуватись під час HTTP-запитів. Вказівники створюємо до кожного шляху.
Функція server.on (“ path”, handle) приймає на вході два значення. Змінна path задає шлях в адресному рядку бровзера, а змінна handle вказує на функцію, котра буде виконуватись під час HTTP запиту.
Список вказівників наведений нижче:

server.on("/", OnConnect) — головна сторінка

server.on("/airtemp", Get_Temperature) — отримати значення температури

server.on("/airpress", Get_Pressure) — отримати значення атмосферного тиску

server.on("/altitude", Get_Altitude) — отримати значення висоти над рівнем моря

server.onNotFound(NotFound) — за відсутності відповіді вивести повідомлення про помилку

Функція sever.begin() - запускає веб-сервер.

Для обробки фактичних HTTP-запитів необхідно викликати функцію server.handleClient() у функції головного циклу loop (). Фрагмент коду головного циклу показаний на рис. 3.12.

```
51
52 void loop() {
53   server.handleClient();
54 }
55
```

Рис. 3.12. Код головного циклу.

Код обробки HTTP-запитів OnConnect, NotFound, а також функції формування головної сторінки показано на рис. 3.13.

```
56 void OnConnect() {
57   temperature = bmp.readTemperature();
58   pressure = bmp.readPressure() * 0.0075;
59   altitude = bmp.readAltitude(SEALEVELPRESSURE_HPA);
60   server.send(200, "text/html", SendHTML(temperature,pressure,altitude));
61 }
62
63 void NotFound(){
64   server.send(404, "text/plain", "ВСЕ ПРОПАЛО !!!");
65 }
66
67 String SendHTML(float temperature, float pressure,float altitude){
68   String  ptr ="<!DOCTYPE html> \n";
69           ptr +="<html>";
70           ptr +="<meta charset=UTF-8>";
71           ptr +="<head>";
72           ptr +="<title>";
73           ptr +="Погодний Чуйник";
74           ptr +="</title>";
75           ptr +="<body>";
76           ptr +="<h1>Погодний чуйник</h1>";
77           ptr +="<p>Температура: ";
78           ptr +=temperature;
79           ptr +="&deg;C</p>";
80           ptr +="<p>Вологість: ";
81           ptr +=humidity;
82           ptr +="%</p>";
83           ptr +="<p>Тиск: ";
84           ptr +=pressure;
85           ptr +="mmHg</p>";
86           ptr +="<p>Висота над р.м.: ";
87           ptr +=altitude;
88           ptr +="m</p>";
89           ptr +="</body>\n";
90           ptr +="</html>\n";
91   return ptr;
92 }
```

Рис. 3.13. Фрагмент коду обробки запитів та формування сторінки

Функція OnConnect отримує значення вимірювань сенсора і формує відповідь на HTTP-запит. Дані вимірювань отримуються викликом відповідних функцій

bmp.readTemperature() — отримати значення температури.

bmp.readPressure() * 0.0075 — отримати значення атмосферного тиску. Для наглядності переведемо значення у міліметри ртутного стовпчика.

bmp.readAltitude(SEALEVELPRESSURE_HPA) — отримати значення вистоти над рівнем моря.

Строкова функція SendHTML отримує на вході значення вимірювань сенсором і повертає HTML код , для головної сторінки. Функція server.send () відповідь на HTML-запит.

Функція NotFound () виводить повідомлення про помилку при неправильно вказаної адреси у бровзері.

Код обробки HTTP-запитів для виводу безпосередніх значень вимірювань наведений на рис. 3.14.

```
94 void Get_Temperature () {
95   static char outemp [6];
96     temperature = bmp.readTemperature();
97     dtostrf(temperature, 2, 2, outemp);
98 
99     server.send(200, "text/plain", outemp);
100 }
101 
102 void Get_Pressure (){
103   static char outpress [8];
104     pressure = bmp.readPressure() * 0.0075;
105     dtostrf(pressure, 3, 2, outpress);
106 
107     server.send(200, "text/plain", outpress);
108 }
109 
110 void Get_Altitude (){
111   static char outalt [8];
112     altitude = bmp.readAltitude(SEALEVELPRESSURE_HPA);
113     dtostrf(altitude, 4,2,outalt);
114 
115     server.send(200, "text/plain", outalt);
116 }
```

Рис. 3.14. Код обробки безпосередніх значень вимірювань.

При звертанні за віповідними шляхами у адресній строці бровзера будуть

виводитись просто числові значення вимірювань. Такий підхід дозволить спростити обробку отриманих даних на клієнтській стороні. Всі функції однакові за структурою і відрізняються тільки змінними яким присвоєні відповідні величини вимірювань. Оскільки серіер надсилає дані у вигляді строки, то отримані дані вимірювань слід із числового типу перетворити у строковий тип. Для цього потрібно створити символьний масив і записати в нього перетворені дані. Перетворення здійснюється викликом функції dstostrf(a,b,c,d). На вході вході функції задаємо чотири змінні

a – змінна числового типу

b – кількість символів до коми

c – кількість символів після коми

d – вихідний символьний массив

Компіляція програми, результати роботи.

Перед основною компіляцією слід перевірити створений код на наявність помилок. Для запуску процесу перевірки необхідно клікнути на іконку у лівому верхньому кутку, як показано на рис. 3.15.

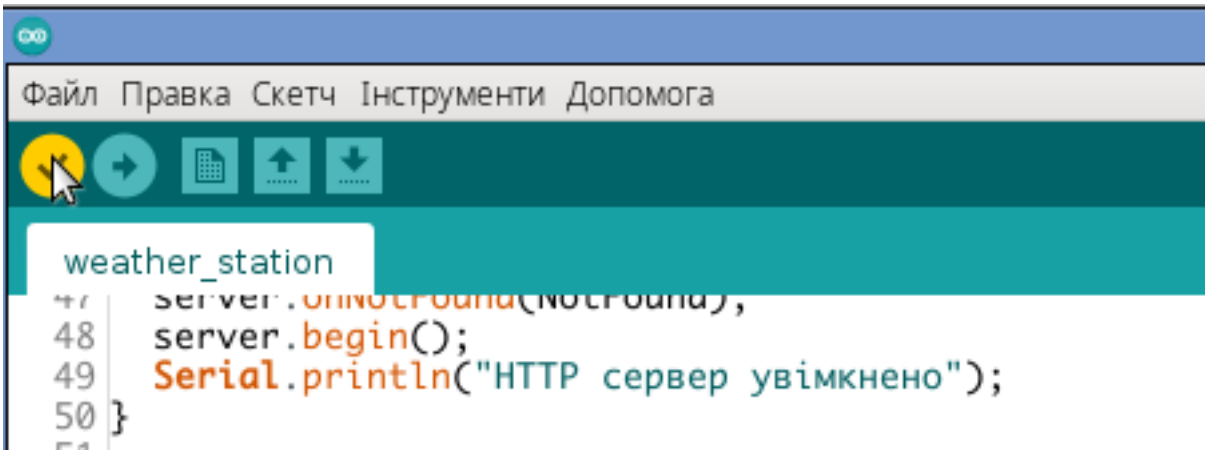

Рис. 3.15. Запуск процесу перевірки коду.

За умови відсутності помилок у терміналі буде виведено повідомлення як показано на рис. 3.16.

```
Змінилися налаштування збирання, все має бути перезібрано
Archiving built core (caching) in: /tmp/arduino_cache_591379/core/core_esp8266_es
. Variables and constants in RAM (global, static), used 29704 / 80192 bytes (37%)
|   SEGMENT  BYTES    DESCRIPTION
|-- DATA     1512     initialized variables
|-- RODATA   1520     constants
`-- BSS      26672    zeroed variables
. Instruction RAM (IRAM_ATTR, ICACHE_RAM_ATTR), used 61167 / 65536 bytes (93%)
|   SEGMENT  BYTES    DESCRIPTION
|-- ICACHE   32768    reserved space for flash instruction cache
`-- IRAM     28399    code in IRAM
. Code in flash (default, ICACHE_FLASH_ATTR), used 284100 / 1048576 bytes (27%)
|   SEGMENT  BYTES    DESCRIPTION
`-- IROM     284100   code in flash
```

Рис. 3.16. Вивід результатів перквірки коду.

Наступним кроком скомпілюємо програму у машинний код а також запишемо готовий бінарний файл до пам’яті мікроконтролера. Програматор мікроконтролера вбудований у пристрій. Такий підхід спрощує процес налагодження, оптимізації та вдосконалення пристрою, а також не потребує додаткового обладнання.

Для запуску процесу компіляції і вивантаження бінарного файлу до мікроконтролера слід натиснути на іконку вивантаження, як показано на рис. 3.17.

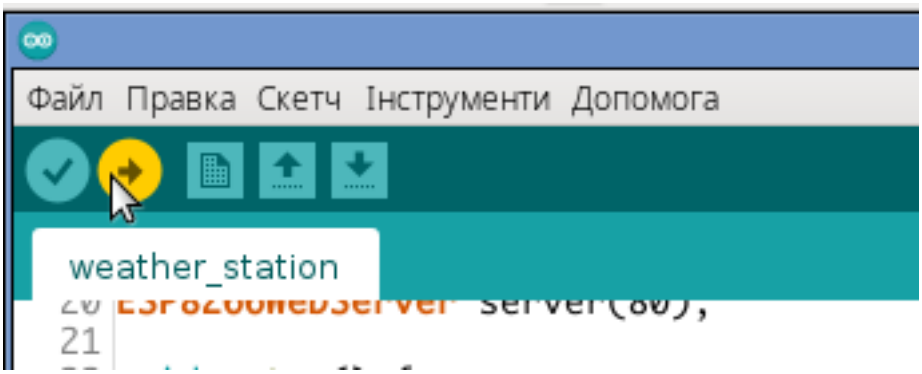

Рис. 3.17. Запуск процесу компіляції і вивантаженя коду до мікроконтролера.

Результат роботи компілятора та програматора мікроконтролера показаний на рис. 3.18.

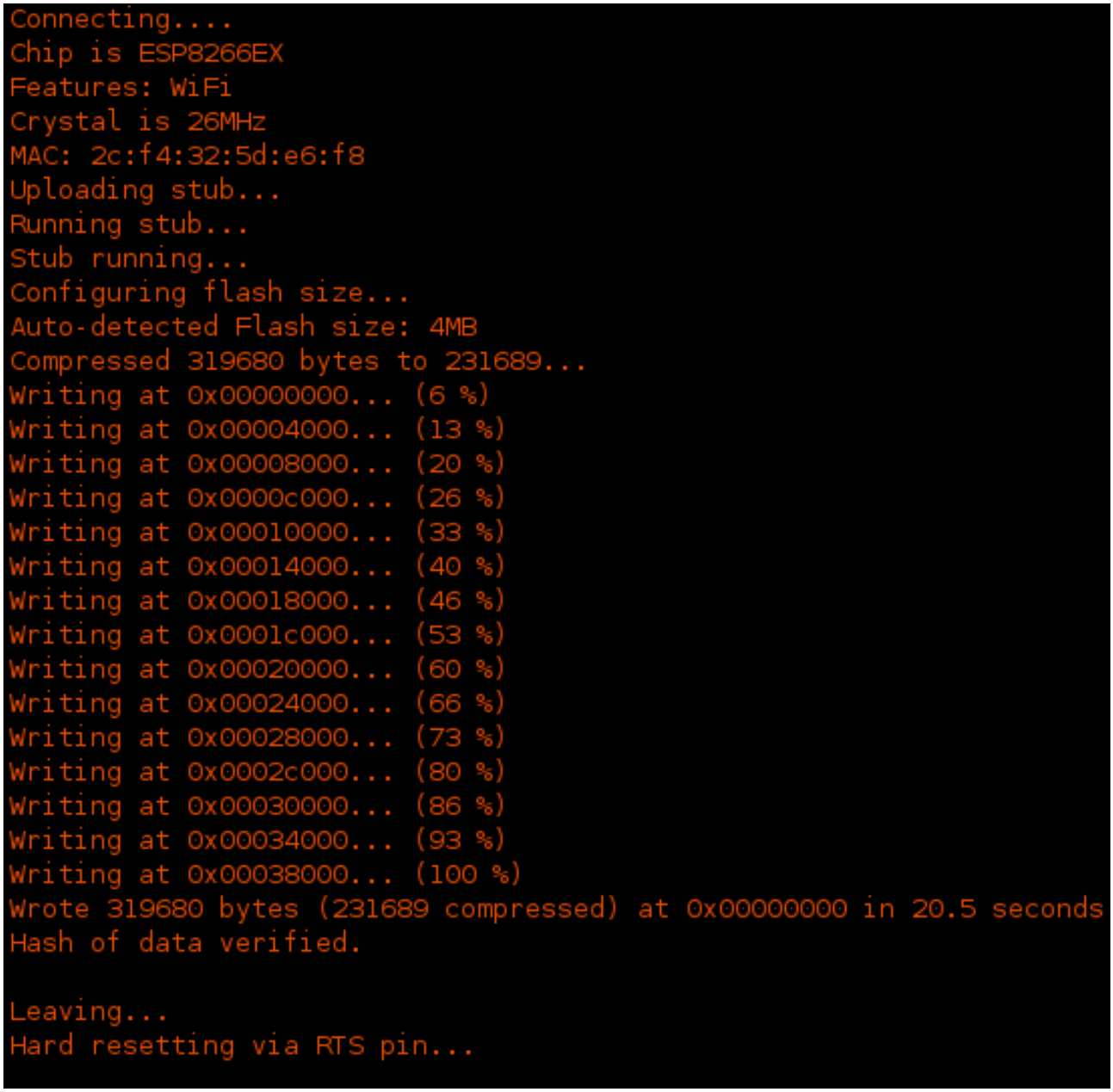

Рис. 3.18. Результат роботи компіляції і вивантаження коду до мікроконтролера.

Після завантаження машинного коду до мікроконтролера для перевірки роботи пристрою інтерфейсу UART пристрою підключимо до комп’ютера. На комп’ютері відкриємо програму-термінал і налаштуємо її підключення до пристрою. Приклад налаштування програми-терміналу показаний на рис. 3.19.

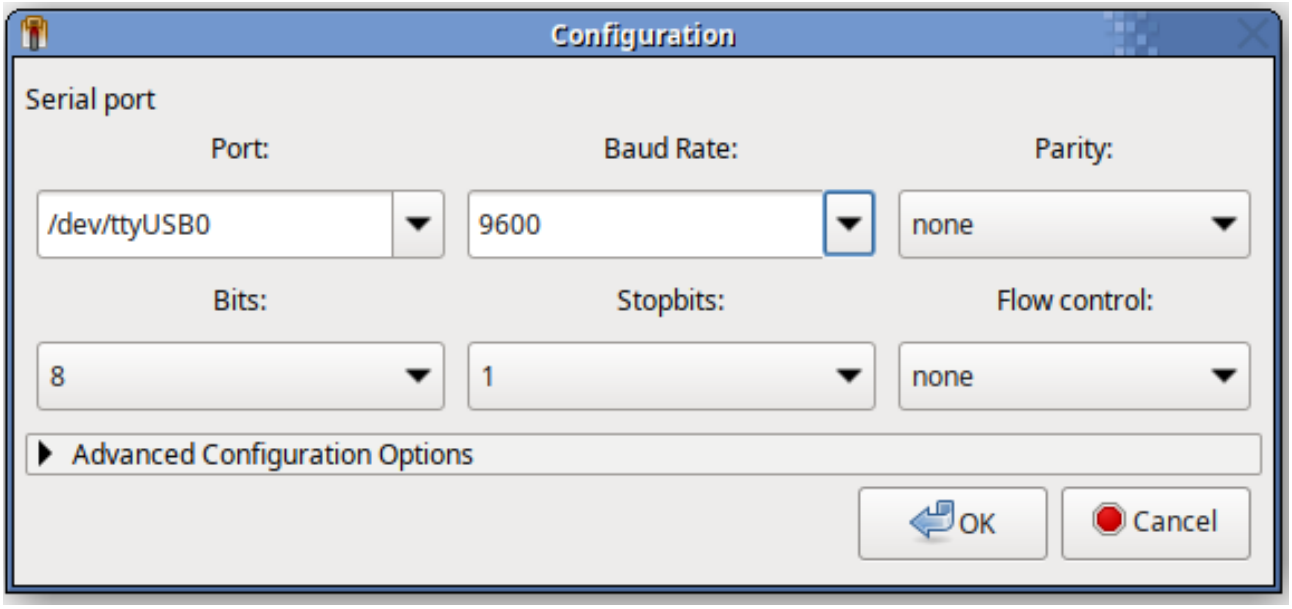

Рис. 3.19. Приклад налаштування програми терміналу.

Наступним кроком вмикаємо живлення пристрою. Прис справності пристрою у вікні терміналу мають вивестись символи ініціалізації вбудованого обладнання мікроконтролера, а також діагностичні повідомлення. Результат запуску пристрою і повідомлення у терміналі показані на рис. 3.20.

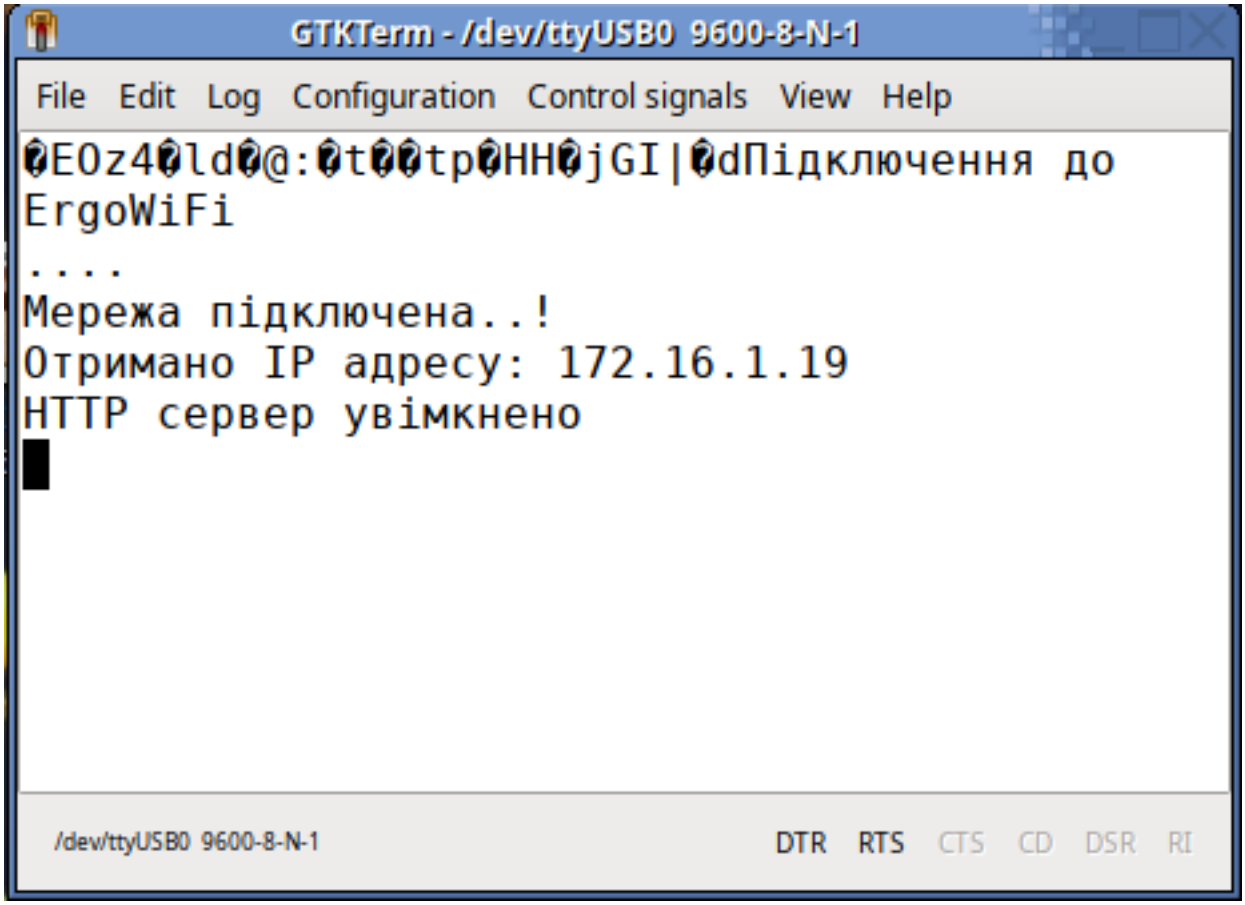

Рис. 3.20. Запуск пристрою із виводом результатів діагностики.

Наступним кроком перевіримо роботу веб-сервера. Для після включення пристрою необхідно у адремну строку бровзера писати IP адресу пристрою. Отримана адреса буде виведена у діагностичному повідомлені на програму-термінал (див. Рис.20). Результат HTTP-запиту показаний на рис. 3.21.

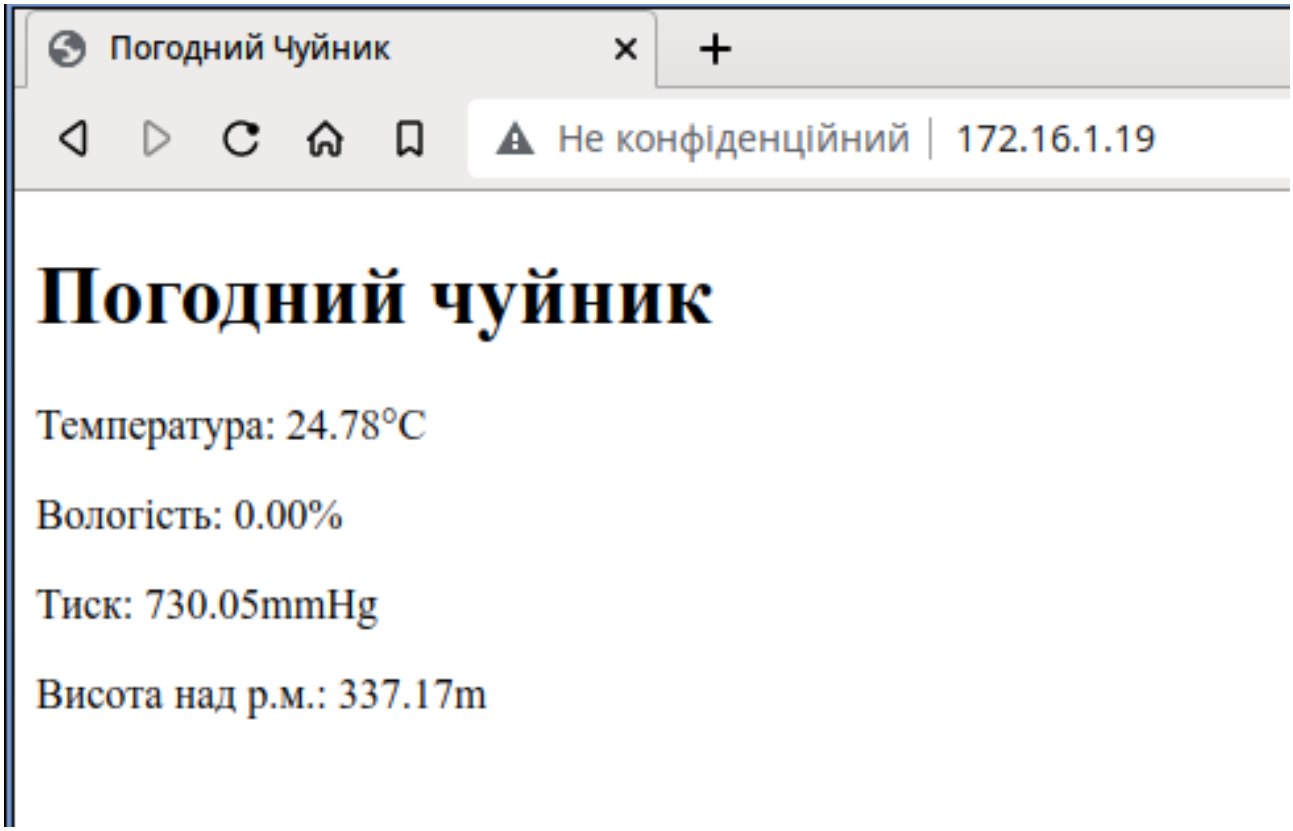

Рис. 3.21. Результат HTTP-запиту.

Для перевірки роботи HTTP-запитів з отриманням безпосередніх значень вимірювань в адресну строку через символ “/” допишемо один із вказівників що описані у вказівниках привязки адреси до функції описаних вище. (див. Рис.3.11)

Результати роботи показані на рис. 3.22 , рис. 3.23, рис. 3.24.

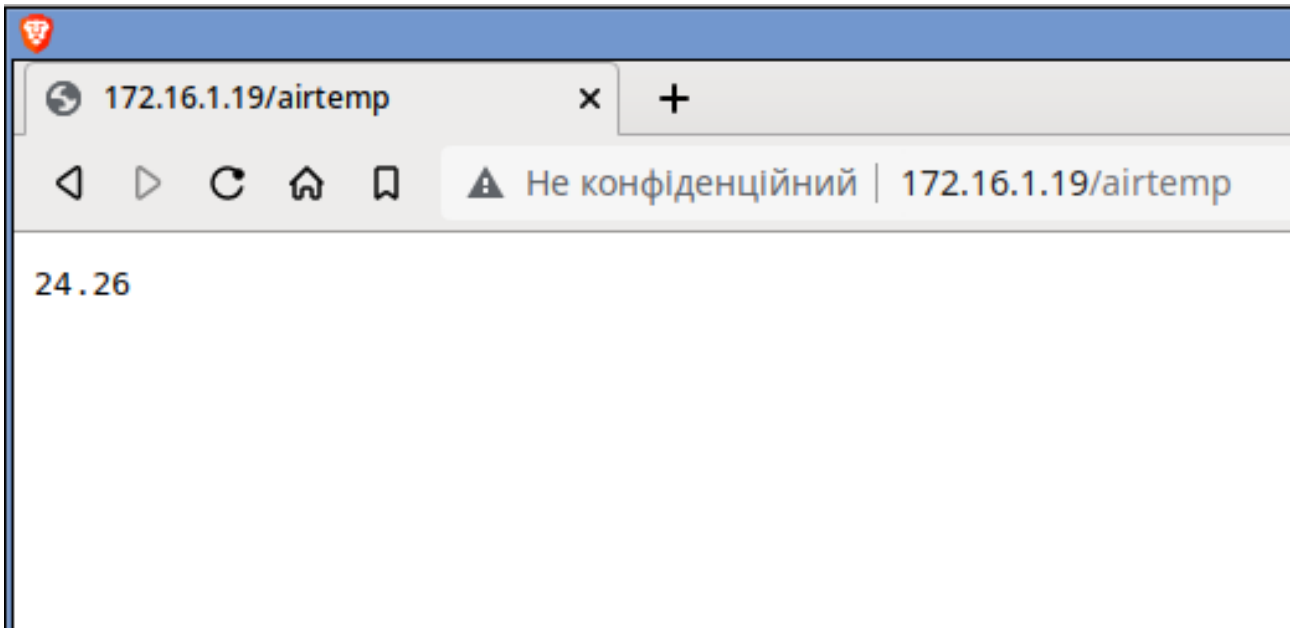

Рис. 3.22. Результат роботи запиту вимірювань температури повітря, за вказівником airtemp.

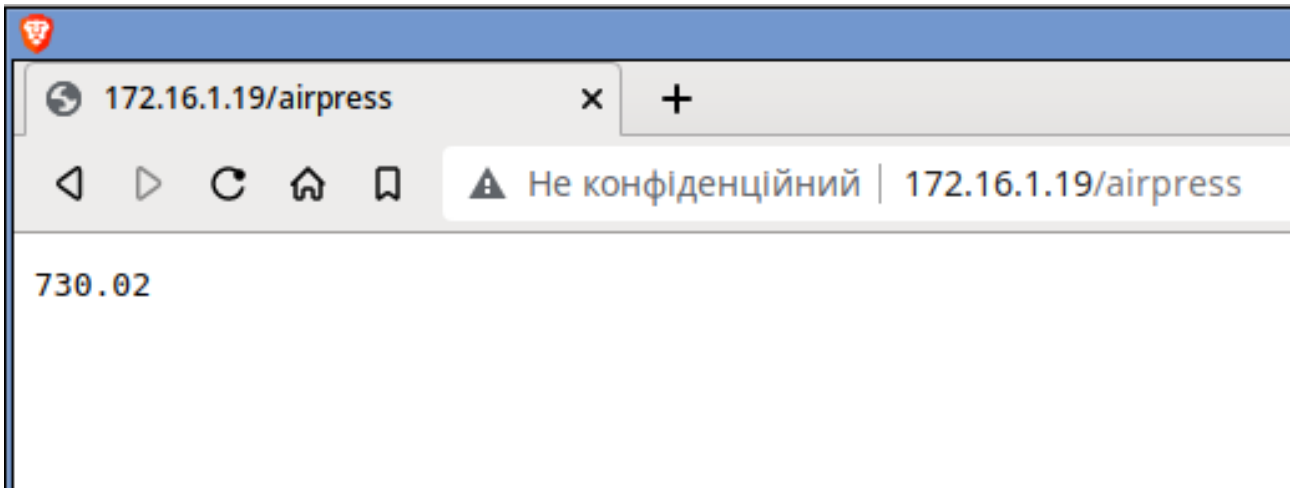

Рис. 3.23. Результат роботи запиту вимірювань атмосферного тиску, за вказівником airpress.

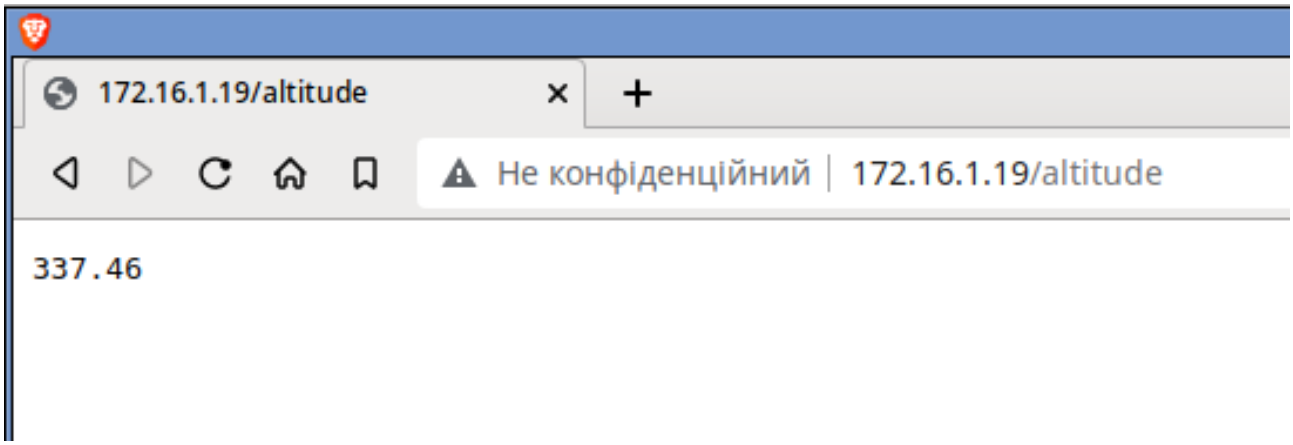

Рис. 3.24. Результат роботи запиту вимірювань вистоти над рівнем моря, за вказівником altitude.

ВИСНОВКИ

Для пристрою створеного на основі мікроконтролера і сенсорів для вимірювання температури та тиску створино програмне забезпечення (прошивку, firmware). Мікроконтролер що використовується у пристрої — ESP8266 фірми Espressif Systems. Сенсора для вимірювань BMP-280 фірми Bosch Sensortec GmbH. Програмне забезпечення надає можливість підключення до мережі Ethernet, зчитує дані вимірювань із сенсорів та дає результати вимірювань за HTTP запитом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Bell D.A. Electronic instrumentation and measurements. - 2nd Ed. -Oxford: Oxford University Press, 2007, 451 p.
2. HelfrickA.D., Cooper W.D. Modern electronic instrumentation and measurement techniques. – London: Prentice-Hall International, 2008. – 446 p.
3. Єрмілова Н.В., Кислиця С.Г. Основи метрології і електричних вимірювань. – Полтава: ПолтНТУ, 2017. - 141 с.
4. Поліщук Є.С. , Дорожовець М.М., Яцук В.О. Метрологія та вимірювальна техніка. - Друге видання. - Львів: Видавництво Львівської політехніки, 2012. 544 с.
5. Метрологія та вимірювання: навчальний посібник / Ю.В. Гнусов, В.В. Тулупов, В.М. Пересічанський; Харк. нац. ун-т внутр. справ, 2019. - 125 с.
6. Barrett S. Arduino I: Getting Started. – Springer, 2020. – 202 p.
7. Barrett S. Arduino II: Systems. – Springer, 2020. – 269 p.
8. Barrett S. Arduino III: Internet of Things. – Springer, 2021. – 217 p.
9. Kurniawan A. Beginning Arduino Nano 33 IoT: Step-By-Step Internet of Things Projects. – Apress, 2021. - 302 p.
10. Kurniawan A. IoT Projects with Arduino Nano 33 BLE Sense. – Apress, 2021. – 295 p.
11. Pulver T. Hands-On Internet of Things with MQTT: Build Connected IoT Devices with Arduino and MQTT. - Packt Publishing, 2019.
12. Prabowo N., Irwanto I. The Implementation of Arduino Microcontroller Boards in Science: A Bibliometric Analysis from 2008 to 2022. – Journal of Engineering Education Transformations, 2023.
13. Pietrosemoli E., Rainone M., Zennaro M. On Extending the Wireless Communications Range of Weather Stations Using LoRaWAN, 2019.

14. Peña-Rodríguez J., Salgado-Meza P. A., Asorey H., Núñez L. A., Núñez-Castiñeyra A., Sarmiento-Cano C., Suárez-Durán M. RACIMO@Bucaramanga: A Citizen Science Project on Data Science and Climate Awareness, 2022.
15. Wenzel M., Brass S. Declarative Programming for Microcontrollers – Datalog on Arduino, 2019.

ДОДАТКИ

```
#include <ESP8266WebServer.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP280.h>

#define SEALEVELPRESSURE_HPA (1013.25)

#define BMP_SCK  (13)
#define BMP_MISO (12)
#define BMP_MOSI (11)
#define BMP_CS   (10)

Adafruit_BMP280 bmp; // підключаємось по шині I2C

float temperature, humidity, pressure, altitude;

/*вписуємо потрібну назву мережі і пароль*/


const char* ssid = "ErgoWiFi";  // назва мережі
const char* password = "energone2020";  //пароль для підключення

ESP8266WebServer server(80);

void setup() {
  Serial.begin(115200);
  delay(100);

  bmp.begin(0x76);

  Serial.println("Підключення до ");
  Serial.println(ssid);

  // запускаємо процес підключення до локальної мережі
  WiFi.begin(ssid, password);

  //перевіряємо підключення
  while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.print(".");
  }
  Serial.println("");
  Serial.println("Мережа підключена..!");
  Serial.print("Отримано IP адресу: ");  Serial.println(WiFi.localIP());

  server.on("/", handle_OnConnect);
  server.on("/airtemp", handle_get_temperature); //прив'язуємо запит по веб адресі до функції
  server.on("/airpress", handle_get_pressure);
  server.on("/altitude", handle_get_altitude);
```

```
  server.onNotFound(handle_NotFound);
  server.begin();
  Serial.println("HTTP сервер увімкнено");
}

void loop() {
  server.handleClient();
}

void handle_OnConnect() {
  temperature = bmp.readTemperature();
  pressure = bmp.readPressure() * 0.0075;
  altitude = bmp.readAltitude(SEALEVELPRESSURE_HPA);
  server.send(200, "text/html", SendHTML(temperature,pressure,altitude));
}

void handle_NotFound(){
  server.send(404, "text/plain", "ВСЕ ПРОПАЛО !!!");
}

String SendHTML(float temperature, float pressure,float altitude){
  String  ptr ="<!DOCTYPE html> \n";
          ptr +="<html>";
          ptr +="<meta charset=UTF-8>";
          ptr +="<head>";
          ptr +="<title>";
          ptr +="Погодний Чуйник";
          ptr +="</title>";
          ptr +="<body>";
          ptr +="<h1>Погодний чуйник</h1>";
          ptr +="<p>Температура: ";
          ptr +=temperature;
          ptr +="&deg;C</p>";
          ptr +="<p>Вологість: ";
          ptr +=humidity;
          ptr +="%</p>";
          ptr +="<p>Тиск: ";
          ptr +=pressure;
          ptr +="mmHg</p>";
          ptr +="<p>Висота над р.м.: ";
          ptr +=altitude;
          ptr +="m</p>";
          ptr +="</body>\n";
          ptr +="</html>\n";
  return ptr;
}

void handle_get_temperature () {
  static char outemp [6];
    temperature = bmp.readTemperature();
    dtostrf(temperature, 2, 2, outemp);
```

```
    server.send(200, "text/plain", outemp);    // возвращаем HTTP-ответ
}

void handle_get_pressure (){
  static char outpress [8];
    pressure = bmp.readPressure() * 0.0075;
    dtostrf(pressure, 3, 2, outpress);

    server.send(200, "text/plain", outpress);
}

void handle_get_altitude (){
  static char outalt [8];
    altitude = bmp.readAltitude(SEALEVELPRESSURE_HPA);
    dtostrf(altitude, 4,2,outalt);

    server.send(200, "text/plain", outalt);
}
```