ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА
(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій
(повне найменування факультету)

Кафедра інформаційних технологій
(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «Бакалавр»

«Програмна реалізація вимірювань, обробки та передавання даних щодо температури та вологості повітря»

(тема кваліфікаційної роботи українською мовою)

«Software implementation of measurements, processing and transmission of data on temperature and humidity»

(тема кваліфікаційної роботи англійською мовою)

Виконав: здобувач заочної форми навчання
спеціальності 122 Комп'ютерні науки
(код, назва спеціальності)
Освітня програма Комп'ютерні науки
(назва)
СІВАШ Павло Андрійович
(прізвище, ім'я, по-батькові здобувача)

Керівник д.т.н., доцент Великодний С.С. _____
(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент к.т.н., доцент кафедри інформаційних технологій НУ ОЮА, Гура В.І.
(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій
№ ___ від ___ _____ 2025 р.

Завідувачка кафедри

_____ КАЗАКОВА Надія
(підпис) (прізвище,ім'я)

Захищено на засіданні ЕК № _____
протокол № ___ від _____ 2025 р.

Оцінка _____ / _____ / _____
(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК

_____ КОПИЧЕНКО Іван
(підпис) (прізвище,ім'я)

Одеса 2025

АНОТАЦІЯ

У кваліфікаційній роботі розробляється тема «Програмна реалізація вимірювань, обробки та передачі інформації щодо температури та вологості».

Мета роботи – розробити функціональну метеостанцію, яка складається з датчиків температури та вологості, що здатні здійснювати вимірювання основних параметрів навколишнього середовища, обробляти та передавати ці дані у зручному форматі.

Для досягнення мети роботи, виконано низку встановлених завдань:

– проаналізовано особливості платформи Arduino;

– складено схему підключення сенсорів;

– розроблено програмне забезпечення для роботи з сенсорами;

– виконано аналіз роботи алгоритму читання даних;

– розширено функціональності метеостанції.

Реалізована система збору, обробки та передачі даних температури і вологості може бути застосована у різних сферах: від побутових метеостанцій до промислових систем моніторингу мікроклімату.

Розроблений проєкт може дозволити виконувати агрометеорологічні вимірювання та сприятиме подальшому розвитку автоматизованих систем моніторингу навколишнього середовища.

Простота реалізації та доступність компонентів дозволяють використовувати систему як навчальний проєкт для вивчення основ роботи з мікроконтролерами та сенсорами. Подібні проєкти є відмінним інструментом для вивчення принципів роботи датчиків та комунікаційних протоколів, що сприяє розвитку технічних навичок у студентів та інженерів-початківців.

ABSTRACT

In the qualification work, the topic "Software implementation for measurement, processing, and transmission of temperature and humidity data" is developed.

The aim of the work is to design a functional weather station consisting of temperature and humidity sensors capable of measuring key environmental parameters, processing the data, and transmitting it in a user-friendly format.

To achieve this goal, the following tasks have been accomplished:

- analyzed the features of the Arduino platform;
- created a connection diagram for the sensors;
- developed software to interact with the sensors;
- analyzed the data reading algorithm;
- extended the weather station's functionality.

The implemented system for collecting, processing, and transmitting temperature and humidity data can be applied in various fields: from household weather stations to industrial microclimate monitoring systems.

The developed project enables agro-meteorological measurements and contributes to the further development of automated environmental monitoring systems.

The simplicity of implementation and the availability of components make this system suitable as an educational project for studying the basics of working with microcontrollers and sensors. Such projects are excellent tools for learning the principles of sensor operation and communication protocols, fostering the development of technical skills in students and novice engineers.

ЗМІСТ

Стор.

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

DHT11 – датчик вологості та температури

GND – ground (заземлення)

GSM – Global System for Mobile Communications (глобальна система мобільного зв'язку)

IDE – Integrated Development Environment (інтегроване середовище розробки)

LED – Light-Emitting Diode (світлодіод)

ВСТУП

У сучасному світі автоматизація та використання інтелектуальних пристроїв відіграють важливу роль у повсякденному житті. Однією з актуальних задач є моніторинг стану навколишнього середовища для забезпечення комфорту та безпеки людини. Саме тому метеостанції набули значного поширення як у промислових, так і в побутових умовах. Вони дозволяють отримувати важливу інформацію про погодні умови:

- температуру;
- вологість повітря;
- атмосферний тиск;
- швидкість та напрям вітру;
- тощо.

Такі дані можуть використовуватися для планування діяльності в різних сферах – від сільського господарства до транспортної логістики та авіації.

У даній дипломній роботі розглядається створення метеостанції на базі платформи Arduino. Вибір Arduino зумовлений її доступністю, гнучкістю та широким функціоналом для роботи з різними датчиками та модулями. Arduino дозволяє створювати недорогі та ефективні рішення для моніторингу погодних умов із можливістю подальшого розширення функціоналу.

Метеостанція на базі Arduino є відмінним прикладом застосування мікроконтролерних систем для автоматизації збору даних про погоду.

Завдяки відкритому програмному середовищу та великій кількості доступних бібліотек і прикладів, розробка подібних пристроїв стає доступною навіть для початківців.

Проєкт передбачає використання різних датчиків для вимірювання температури, вологості, атмосферного тиску та інших параметрів, а також передачу зібраних даних на локальний дисплей для їх подальшого аналізу.

Однією з головних переваг даного проєкту є можливість адаптації метеостанції під конкретні потреби користувача. Наприклад, можна додати датчики для вимірювання рівня ультрафіолетового випромінювання, рівня опадів чи якості повітря. Також важливим аспектом є можливість інтеграції метеостанції з іншими системами, такими як «розумний дім», або передача даних через інтернет для віддаленого моніторингу.

Використання метеостанції на базі Arduino дозволяє значно знизити вартість проєкту порівняно з комерційними аналогами, не втрачаючи при цьому точності та надійності вимірювань. Крім того, такі проєкти є відмінним інструментом для вивчення принципів роботи мікроконтролерів, датчиків та комунікаційних протоколів, що сприяє розвитку технічних навичок у студентів та інженерів-початківців.

1 АНАЛІЗ ОБРАНОЇ ПРЕДМЕТНОЇ ГАЛУЗІ ЗАСТОСУВАННЯ ARDUINO

1.1 Основні компоненти плати

Ардуіно (Arduino) – це популярна платформа для створення електронних пристроїв, яка поєднує апаратну частину (мікроконтролер) та програмне забезпечення. Arduino використовується як новачками, так і професіоналами для створення різних проєктів: від простих пристроїв, таких як світлодіодні лампи, до складних роботів та автоматичних систем [1].

На базі Arduino можна виконувати програмування для виконання різних завдань, таких як керування датчиками, двигунами та іншими компонентами. Основною перевагою Arduino є його простота використання та велика спільнота розробників.

Основними компонентами Arduino є [2]:

– мікроконтролер – це «мозок» плати, який виконує всі команди (наприклад, ATmega328 у платі Arduino Uno);

– цифрові та аналогові входи / виходи (GPIO) – для підключення різних пристроїв, датчиків та модулів;

– порти живлення – для підключення зовнішнього джерела енергії;

– USB-порт – для програмування плати через комп’ютер;

– світлодіоди (LED) – для індикації роботи плати.

Найпопулярнішими платами Arduino є [3]:

– Arduino Uno – найпоширеніша плата, підходить для початківців;

– Arduino Nano – компактна плата для проєктів з обмеженим простором;

– Arduino Mega – має більше входів/виходів, підходить для складних проєктів;

– Arduino Leonardo – має можливість емулювати клавіатуру чи мишу.

1.2 Можливості Arduino

Проаналізуємо, що можна створити за допомогою Arduino [4]:

– роботів;

– автоматизовані системи для «розумного дому»;

– метеостанції;

– 3D-принтери;

– музичні інструменти;

– іграшки та інтерактивні пристрої.

Сфокусуємо нашу увагу на створенні автоматизованої метеостанції, як раз яка буде об’єктом кваліфікаційної роботи.

Що ж можна винести до переваг Arduino – це:

– простота у використанні;

– відкрите програмне забезпечення та апаратне забезпечення;

– велика спільнота розробників;

– багато готових прикладів та бібліотек.

1.3 Огляд датчиків вологості та температури для Arduino

Датчики вологості та температури є важливими компонентами для створення метеостанцій. Вони дозволяють отримувати ключові дані про стан навколишнього середовища, що є необхідним для аналізу кліматичних умов. У цьому підрозділі буде розглянуто основні типи датчиків, які найчастіше використовуються з платформою Arduino.

1.3.1 Датчики температури

1. Датчик DS18B20 – це цифровий датчик температури з високою точністю. Він підтримує комунікацію за допомогою протоколу 1-Wire, що дозволяє підключати кілька датчиків до одного цифрового піна Arduino.

Характеристики [5]:

– діапазон вимірювань: від мінус 55°C до +125°C;

– точність: ± 0,5°C у діапазоні від мінус 10°C до +85°C;

– інтерфейс: 1-Wire;

– можливість роботи на великій відстані від контролера.

2. Датчик TMP36 – аналоговий датчик температури, який видає напругу пропорційну температурі. Він легко підключається до аналогового входу Arduino.

Характеристики [5]:

– діапазон вимірювань: від мінус 40°C до +125°C;

– точність: ±1°C;

– напруга живлення: від 2,7 В до 5,5 В;

– простий у використанні.

1.3.2 Датчики вологості

1. Датчик DHT11 – це популярний датчик, який може вимірювати як температуру, так і вологість повітря. Він забезпечує базову точність і є ідеальним вибором для простих проєктів.

Характеристики [5]:

– діапазон вимірювань температури: від 0°C до +50°C;

– діапазон вимірювань вологості: від 20% до 90% RH;

– точність: ±2°C та ±5% RH;

– цифровий інтерфейс.

2. Датчик DHT22 (AM2302) – є вдосконаленою версією DHT11 і забезпечує більш високу точність вимірювань.

Характеристики [5]:

– діапазон вимірювань температури: від мінус 40°C до +80°C;

– діапазон вимірювань вологості: від 0% до 100% RH;

– точність: ±0,5°C та ±2% RH;

– цифровий інтерфейс.

1.3.3 Порівняння датчиків

Вибір конкретного датчика залежить від:

– вимог проєкту щодо точності;

– діапазону вимірювань;

– умов експлуатації.

Arduino забезпечує просте підключення та взаємодію з більшістю популярних датчиків, що робить його універсальним рішенням для створення метеостанцій. Датчики температури та вологості, які можуть використовуватись для створення метеостанції зведено до табл. 1.1.

Таблиця 1.1 – Датчики температури та вологості, які використовуються для створення метеостанції

Датчик	Діапазон температур:	Точність температури:	Діапазон вологості:	Точність вологості:	Інтерфейс:
DS18B20	-55°C до +125°C	±0,5°C	–	–	1-Wire
TMP36	-40°C до +125°C	±1°C	–	–	Аналоговий
DHT11	0°C до +50°C	±2°C	20% до 90% RH	±5% RH	Цифровий
DHT22	-40°C до +80°C	±0,5°C	0% до 100% RH	±2% RH	Цифровий

1.4 Аналіз середовища програмування для Arduino

Arduino IDE (Integrated Development Environment) є основним середовищем для розробки програмного забезпечення для платформ Arduino.

Це зручне, інтуїтивно зрозуміле середовище, яке використовується як новачками, так і досвідченими розробниками для створення програм, що працюють на мікроконтролерах Arduino. Середовище надає всі необхідні інструменти для написання, редагування, компіляції та завантаження коду на плату [6].

1.4.1 Основні компоненти Arduino IDE

Проаналізуємо основні компоненти Arduino IDE [7].

а) редактор коду (рис. 1.2):

– забезпечує написання скетчів (програм) мовою C/C++;

– містить базові функції, такі як підсвічування синтаксису, автозавершення команд, зручна навігація;

– проста структура дозволяє новачкам швидко опанувати базові навички програмування;

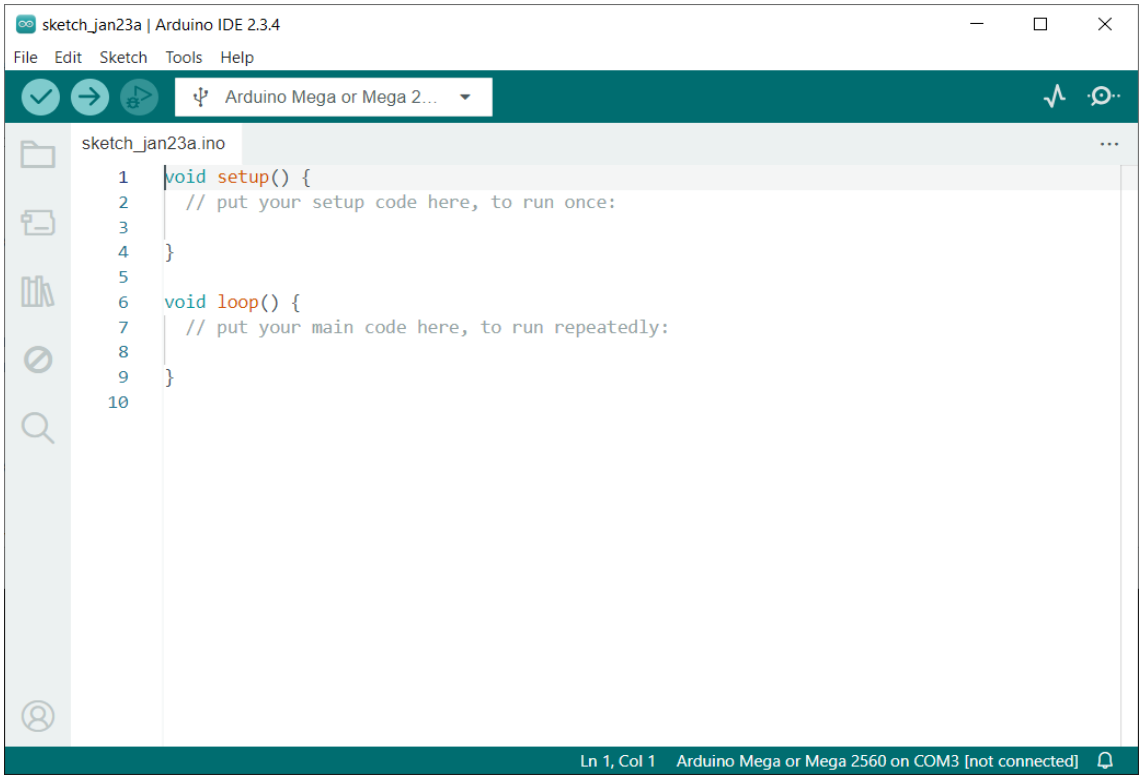

Рисунок 1.2 – Загальний вигляд редактору коду Arduino IDE

б) компілятор та завантажувач:

– використовується компілятор AVR-GCC для перетворення коду у форму, яку розуміє мікроконтролер;

– інструмент avrdude завантажує скомпільований код у пам’ять мікроконтролера через USB-з’єднання;

в) бібліотеки:

– Arduino IDE має вбудовану систему керування бібліотеками, що розширюють функціональність та спрощують роботу з датчиками, дисплеями, двигунами тощо;

– можливість додавання зовнішніх бібліотек робить середовище універсальним для роботи з широким спектром периферійних пристроїв;

г) консоль виводу:

– показує повідомлення компілятора, помилки, попередження та успішне завершення завантаження коду;

– також дозволяє переглядати дані, які надходять з Arduino-плати через послідовний порт;

д) серійний монітор:

– використовується для взаємодії з платою Arduino у реальному часі;

– дає змогу отримувати дані від датчиків чи відправляти команди на плату.

1.4.2 Мова програмування

Програмування у Arduino IDE базується на мові C/C++ з використанням спеціальної бібліотеки Arduino Core.

Ця бібліотека спрощує написання коду за рахунок готових функцій, які забезпечують роботу з апаратними компонентами, такими як:

– цифрові порти;

– аналогові порти;

– таймери;

– послідовні порти;

– тощо.

Основні переваги цієї мови:

– простота у використанні: для виконання багатьох завдань достатньо лише кількох рядків коду;

– гнучкість: підтримка складніших функцій C++ дає змогу створювати оптимізовані програми.

1.4.3 Особливості та переваги Arduino IDE

Проаналізуємо та окремо виділимо особливості та переваги Arduino IDE:

а) кросплатформність: Arduino IDE підтримує Windows, macOS та Linux, що забезпечує доступність для користувачів із різними операційними системами;

б) відкритий код (open-source): Arduino IDE – це програмне забезпечення з відкритим вихідним кодом, що дозволяє спільноті розробників вносити покращення, створювати власні версії та розширення;

в) зручність для новачків: IDE розроблено з урахуванням користувачів без досвіду програмування, що робить процес навчання швидким і зрозумілим;

г) розширення функціоналу: окрім офіційного Arduino IDE, існують альтернативні середовища, такі як PlatformIO чи Visual Studio Code із плагіном для Arduino, які забезпечують розширений функціонал;

д) велика спільнота: завдяки широкій популярності Arduino, користувачі мають доступ до величезної кількості:

– навчальних матеріалів;

– документації;

– форумів;
– прикладів коду.

1.4.4 Обмеження та недоліки Arduino IDE

Проаналізуємо та окремо виділимо обмеження та недоліки Arduino IDE:

а) відсутність розвинених інструментів для рефакторингу коду чи інтегрованого відлагодження;

б) інтерфейс може здатися надто простим для досвідчених розробників, які працюють зі складними проєктами;

в) для великих проєктів можуть знадобитися альтернативні середовища розробки із кращою інтеграцією інструментів.

1.5 Висновки до розділу та постановка завдання

При виконанні аналізу обраної предметної галузі, можна скласти наступні висновки та сформулювати мету проєкту.

Застосування Arduino IDE є ідеальним вибором для розробки проєктів із використанням платформи Arduino, таких як цифрова метеостанція.

Простота використання, доступність та підтримка спільноти роблять платформу Arduino чудовим інструментом для навчання основ програмування мікроконтролерів та реалізації прототипів. Однак для масштабних проєктів можуть знадобитися більш розвинені середовища розробки.

Мета даного проєкту – розробити функціональну метеостанцію, яка складається з датчиків температури та вологості, що здатні здійснювати вимірювання основних параметрів навколишнього середовища, обробляти та передавати ці дані у зручному форматі.

Для досягнення мети роботи, необхідно виконати низку встановлених завдань:

- проаналізувати особливості платформи Arduino;
- скласти схему підключення сенсора DHT11;
- розробити програмне забезпечення для роботи з DHT11;
- виконати аналіз роботи алгоритму читання даних;
- розширити функціональності метеостанції.

У якості обладнання слід використати платформу Arduino, датчики температури та вологості, а також програмний інтерфейс для передачі зібраних даних користувачу.

2 ОБҐРУНТУВАННЯ ПРОЕКТНОГО РІШЕННЯ ЩОДО АПАРАТНОЇ ТА СХЕМНОЇ РЕАЛІЗАЦІЇ

2.1 Обрання базової плати

У якості базової плати – оберемо плату, яка є у наявності у лабораторії дипломного керівника – це плата Arduino Mega 2560. Надамо технічні характеристики щодо неї.

Arduino Mega 2560 – це одна з найбільш популярних і потужних плат Arduino, яка використовується для реалізації складних проєктів, що потребують великої кількості входів / виходів або значного обсягу пам’яті. Її технічні характеристики роблять її ідеальним вибором для метеостанцій із розширеним функціоналом [8].

2.1.1 Основні характеристики плати

Загальний вигляд плати із роз’ємами наведений на рис. 2.1.

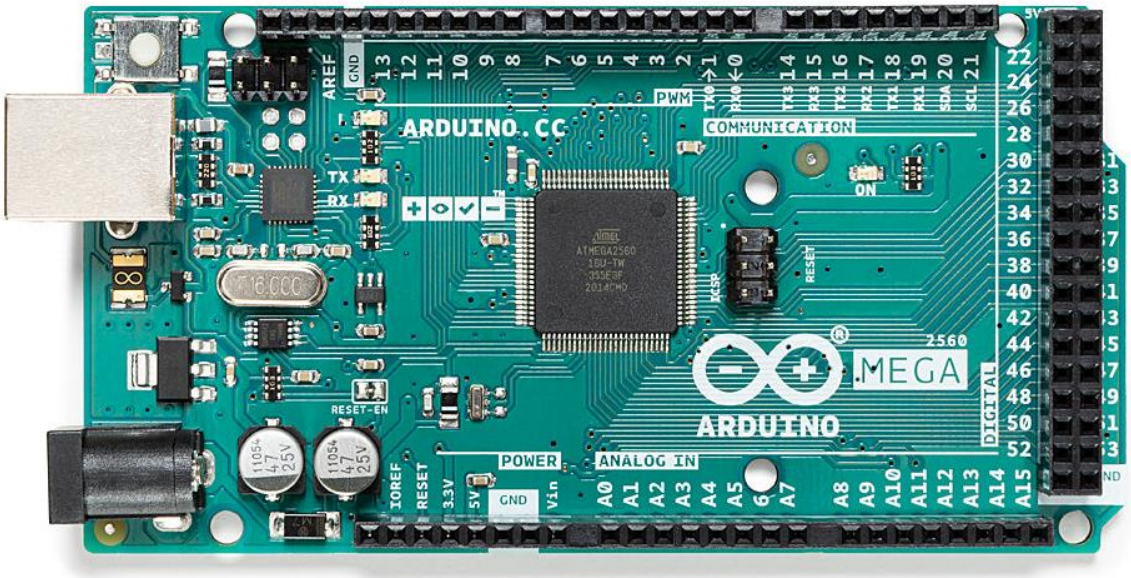

Рисунок 2.1 – Загальний вигляд плати Arduino Mega 2560 із роз’ємами

Загальні технічні характеристики плати наведено у табл. 2.1.

Таблиця 2.1 – Технічні характеристики плати Arduino Mega 2560

Характеристика	Параметр
Тактова частота:	16 МГц
Флеш-пам'ять:	256 КБ
SRAM:	8 КБ
EEPROM:	4 КБ
Кількість цифрових входів / виходів:	54 (із них 15 підтримують PWM)
Кількість аналогових входів:	16
Інтерфейси:	UART, I2C, SPI
Роз'єми живлення:	USB, роз'єм для зовнішнього адаптера (7 – 12 В)
Довжина:	102 мм
Ширина:	53 мм
Вага:	37 г

2.1.2 Переваги

Виділемо основні переваги використання плати Arduino Mega 2560, стосовно до цілей нашого проєкту [9]:

а) розширена кількість пінів: велика кількість цифрових і аналогових входів / виходів дозволяє підключати багато датчиків і модулів одночасно;

б) велика пам'ять: значний обсяг флеш-пам'яті та SRAM забезпечують можливість використання складних програм і збереження великих обсягів даних;

в) підтримка різних інтерфейсів: наявність UART, I2C і SPI робить плату універсальною для підключення периферійних пристроїв;

г) стабільність роботи: Arduino Mega 2560 має покращену систему живлення, що дозволяє уникати збоїв під час роботи з великою кількістю компонентів.

2.1.3 Застосування у метеостанціях

Arduino Mega 2560 ідеально підходить для складних метеостанцій, які потребують інтеграції великої кількості датчиків:

- температури;
- вологості;
- тиску;
- якості повітря;
- тощо.

Завдяки своїй універсальності, Arduino Mega 2560 є чудовим вибором для метеостанцій [10], завдяки своїм технічним характеристикам, а саме:

а) велика кількість пінів: метеостанції часто потребують підключення численних датчиків (температури, вологості, атмосферного тиску тощо) і модулів (дисплеї, модулі передачі даних), 54 цифрових входи / виходи та 16 аналогових входів дозволяють легко підключити всі необхідні компоненти:

б) об’ємна флеш-пам’ять: 256 КБ пам’яті дає змогу створювати складні програми з обробкою даних, інтеграцією з інтернет-сервісами та збереженням журналів вимірювань;

в) можливість масштабування: завдяки великій кількості пінів і достатньому обсягу пам’яті, плата ідеально підходить для розширення функціоналу метеостанції, наприклад, додавання нових датчиків чи інтеграції зі «смарт» системами;

г) сумісність з модулями: Arduino Mega 2560 підтримує численні модулі для передачі даних, включаючи Wi-Fi, Bluetooth і GSM, що є важливим для віддаленого моніторингу;

д) зручність програмування: плата використовує відкриту платформу Arduino IDE, яка підтримує безліч бібліотек для роботи з популярними датчиками та модулями.

Завдяки цим характеристикам, Arduino Mega 2560 забезпечує надійність, гнучкість і масштабованість, необхідні для створення складних і функціональних метеостанцій.

2.2 Аналіз схеми підключення пристроїв

Схема підключення Arduino до датчика DHT11 для вимірювання температури та вологості приведена на рис 2.2.

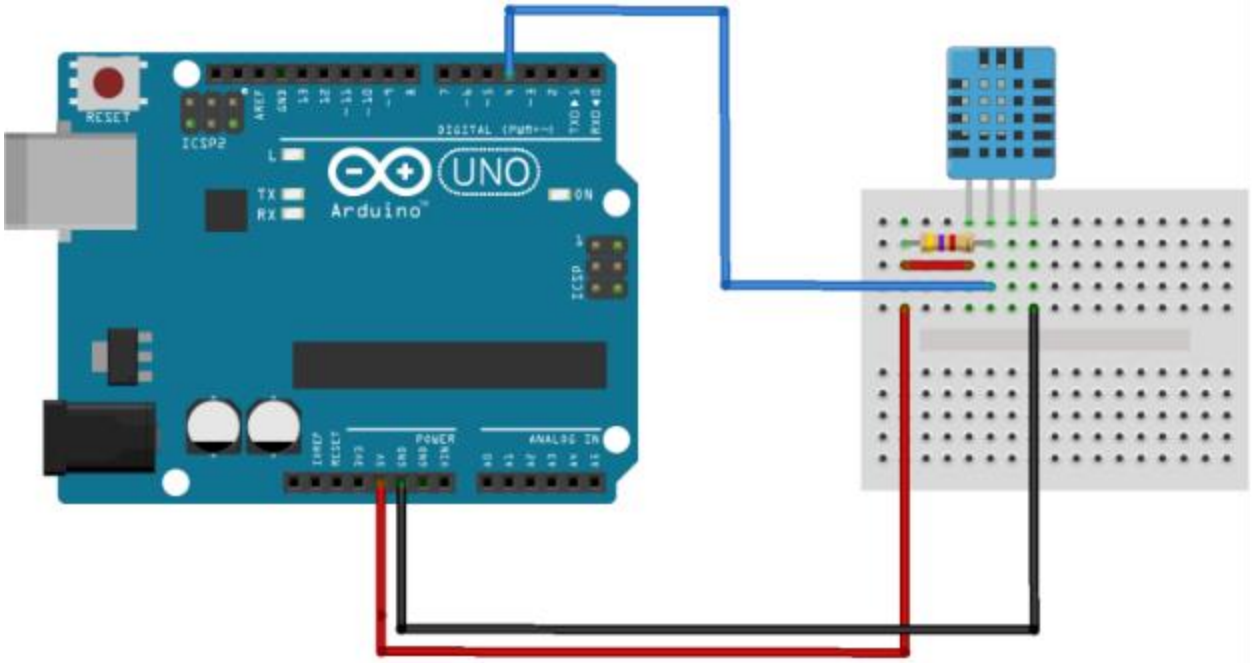

Рисунок 2.2 – Схема підключення Arduino Uno до датчика DHT11

Опишемо її:

а) Arduino Uno: використовується як центральний контролер для обробки сигналів від датчика DHT11;

б) макетна плата: служить для зручного з'єднання компонентів та додавання резисторів.

в) датчик DHT11 – має чотири контакти:

– VCC (живлення);

– DATA (сигнальний контакт) – передає цифрові значення температури та вологості до Arduino;

– N/C (не підключений);

– GND (заземлення).

г) підключення датчика DHT11 до Arduino:

– VCC (живлення) підключено до 5В Arduino;

– DATA (сигнальний контакт) підключено до цифрового піну «4» Arduino через резистор (pull-up) на 10 кОм, який стабілізує сигнал і забезпечує правильну роботу;

– GND (заземлення) підключено до GND Arduino;

д) резистор, номіналом опору 10 кОм, підключаємо між пінами VCC (живлення) і DATA (сигнальний контакт); він використовується для підтягування сигнальної лінії до рівня живлення.

Підтягування сигнальної лінії до рівня живлення (pull-up) означає, що резистор використовується для встановлення логічного рівня «1» на сигнальній лінії, коли інші пристрої не активні. Це забезпечує правильну роботу цифрових входів і виходів у мікроконтролерах та інших цифрових схемах.

Коли сигнальна лінія неактивна (тобто не передає сигнал), її стан може бути «плаваючим» (floating), і вона може приймати випадкові значення через вплив електромагнітного шуму або невизначений стан схеми. Щоб уникнути цього, використовується резистор, який «підтягує» лінію до рівня живлення (логічної «1»), коли інші пристрої її не активують.

Тобто, один кінець резистора підключений до живлення (+ 5 В у випадку Arduino), а інший – до сигнальної лінії. У стані спокою (коли датчик DHT11 нічого не передає), через резистор на лінії встановлюється рівень

логічної «1». Коли датчик починає передавати дані, він може «перетягнути» лінію на логічний рівень «0», активуючи передачу сигналу.

У такому разі забезпечується стабільність сигналу: навіть якщо датчик DHT11 неактивний, то мікроконтролер «бачить» стабільний рівень «1» та усуває невизначений стан на лінії, який може спричинити помилки у передачі даних.

У схемі, наведеної на рис. 2.2, коли датчик DHT11 не активний, сигнальна лінія завдяки резистору підтягнута до + 5 В (логічної «1»). Коли датчик передає дані, він з'єднує сигнальну лінію із землею (GND), формуючи імпульси логічного рівня «0».

Отже, pull-up резистор – це простий, але ефективний спосіб забезпечити правильну роботу цифрових сигналів у схемах із мікроконтролерами.

3 ПРАКТИЧНА ЧАСТИНА ЗАСТОСУВАННЯ ТА РОЗРОБКИ СПЕЦІАЛЬНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Arduino програмується за допомогою спеціального програмного забезпечення – Arduino IDE (Integrated Development Environment). Програми для Arduino називаються скетчами і пишуться на мові програмування, яка базується на C / C++ [11].

3.1 Підключення спеціалізованих бібліотек

Спочатку треба підключити додаткові бібліотеки для забезпечення роботи датчиків. Розберемо ці файли.

3.1.1 Розбір коду заголовочного файлу бібліотеки

Розберемо структуру наступного елементу бібліотеки, що підключається. Цей код є заголовочним файлом бібліотеки для роботи з датчиком DHT11, що вимірює температуру і вологість [12]. Він визначає інтерфейс класу `dht11` і містить константи, макроси та методи для роботи з цим датчиком.

Умови включення бібліотек:

```
#ifndef dht11_h
#define dht11_h
```

Ця секція запобігає повторному включенню цього заголовочного файлу в програму. Якщо `dht11_h` ще не визначено, то файл буде підключений; інакше – ігнорується.

Підключення потрібних бібліотек:

```
#if defined(ARDUINO) && (ARDUINO >= 100)
```

```
#include <Arduino.h>
#else
#include <WProgram.h>
#endif
```

Для підтримки різних версій Arduino IDE:

– у версіях IDE 1.0 і вище використовується бібліотека `<Arduino.h>`;

– для старіших версій IDE використовується `<WProgram.h>`.

Версія бібліотеки:

```
#define DHT11LIB_VERSION "0.4.1"
```

Макрос визначає версію бібліотеки як 0.4.1.

Коди повернення:

```
#define DHTLIB_OK                    0
#define DHTLIB_ERROR_CHECKSUM   -1
#define DHTLIB_ERROR_TIMEOUT    -2
```

Ці макроси визначають коди, що повертаються методом `read(int pin)`:

– `DHTLIB_OK (0)`: дані зчитано успішно;

– `DHTLIB_ERROR_CHECKSUM (-1)`: виявлено помилку контрольної суми;

– `DHTLIB_ERROR_TIMEOUT (-2)`: перевищено час очікування відповіді від датчика.

Оголошення класу `dht11`:

```
class dht11
{
public:
    int read(int pin);
      int humidity;
```

```
      int temperature;
};
```

Клас `dht11` надає інтерфейс для роботи з датчиком.

`read(int pin)` – метод, який ініціює зчитування даних з датчика. Параметр `pin` визначає номер піну, до якого підключений датчик. Повертає один із кодів повернення (`DHTLIB_OK`, `DHTLIB_ERROR_CHECKSUM`, `DHTLIB_ERROR_TIMEOUT`).

`int humidity` - поле для зберігання значення вологості (%).

`int temperature` - поле для зберігання значення температури (°C).

Закриття інструкції умовного включення:

```
#endif
```

Закриває директиву `#ifndef dht11_h`.

Коментар:

```
// END OF FILE
```

Позначає кінець файлу.

Таким чином, цей розібраний заголовочний файл визначає структуру класу `dht11` для взаємодії з датчиком DHT11, а саме:

– забезпечує можливість зчитування температури та вологості;

– використовує коди повернення для визначення результатів роботи;

– підтримує сумісність з різними версіями Arduino IDE.

Це основа бібліотеки, яка використовується в поєднанні з реалізацією у файлі `dht11.cpp`. Розберемо далі саме цей файл.

3.1.2 Розбір коду файлу бібліотеки для роботи з датчиком

Загальна інформація про файл:

```
// FILE: dht11.cpp
// PURPOSE: DHT11 Temperature & Humidity Sensor library for
Arduino
```

Цей файл є бібліотекою для роботи з датчиком DHT11, який вимірює температуру та вологість.

Основні оголошення:

```
#include "dht11.h"
```

Підключення заголовочного файлу, який містить визначення класу `dht11` та основних змінних.

Повертає коди помилок:

```
// Return values:
// DHTLIB_OK
// DHTLIB_ERROR_CHECKSUM
// DHTLIB_ERROR_TIMEOUT
```

Це коментарі, які пояснюють можливі значення, що повертаються функцією `read`:

– `DHTLIB_OK`: дані успішно зчитані;

– `DHTLIB_ERROR_CHECKSUM`: помилка контрольної суми;

– `DHTLIB_ERROR_TIMEOUT`: таймаут під час очікування сигналу.

Функція `read(int pin)`– функція зчитує дані з датчика, підключеного до певного цифрового піну.

Оголошення змінних:

```
uint8_t bits[5];
uint8_t cnt = 7;
uint8_t idx = 0;
```

– bits[5]: буфер для зберігання отриманих 40 біт (5 байтів) даних;

– cnt: лічильник для визначення позиції бітів у байті;

– idx: індекс для поточного байта в буфері.

Очищення буфера:

```
for (int i=0; i< 5; i++) bits[i] = 0;
```

Ініціалізація буфера нулями перед отриманням нових даних.

Запит на зчитування:

```
pinMode(pin, OUTPUT);
digitalWrite(pin, LOW);
delay(18);
digitalWrite(pin, HIGH);
delayMicroseconds(40);
pinMode(pin, INPUT);
```

– встановлюється режим виводу для піну;

– лінія даних утримується низьким рівнем 18 мс для ініціації запиту;

– потім встановлюється високий рівень на 40 мкс, після чого піни переводяться в режим входу для очікування відповіді.

Очікування відповіді датчика:

```
unsigned int loopCnt = 10000;
while(digitalRead(pin) == LOW)
      if (loopCnt-- == 0) return DHTLIB_ERROR_TIMEOUT;

loopCnt = 10000;
while(digitalRead(pin) == HIGH)
      if (loopCnt-- == 0) return DHTLIB_ERROR_TIMEOUT;
```

– датчик відповідає низьким і високим рівнем;

– якщо таймаут перевищено, функція повертає `DHTLIB_ERROR_TIMEOUT`.

Далі йде цикл зчитування даних (40 біт). Цей цикл виконує основну задачу зчитування 40 біт даних від датчика DHT11. Датчик передає дані у вигляді серії високих та низьких сигналів, кожен із яких має певну тривалість. Тривалість сигналу визначає, чи це «0» чи «1».

```
for (int i=0; i<40; i++)
```

– `int i = 0` – ініціалізує змінну-лічильник `i`, яка відповідає за номер біта (від 0 до 39);

– `i < 40` – цикл триває до зчитування всіх 40 бітів;

– `i++` - після кожної ітерації збільшує значення `i` на 1.

Очікування сигналу низького рівня (LOW):

```
loopCnt = 10000;
while(digitalRead(pin) == LOW)
    if (loopCnt-- == 0) return DHTLIB_ERROR_TIMEOUT;
```

`loopCnt = 10000` – ініціалізує лічильник `loopCnt`, щоб уникнути зациклення. Це захист від помилок або зависань, якщо датчик не відповідає.

`while (digitalRead(pin) == LOW)` – чекає, поки сигнал на піні стане високим (HIGH).

`digitalRead(pin)` зчитує поточний стан піна: LOW (0 В) або HIGH (близько 5 В).

`if (loopCnt-- == 0)` – якщо лічильник доходить до 0 (тобто цикл тривав занадто довго), повертається помилка DHTLIB_ERROR_TIMEOUT.

Фіксація часу переходу до сигналу високого рівня:

```
unsigned long t = micros();
```

`micros()` – фіксує поточний час у мікросекундах. Це потрібно, щоб виміряти тривалість сигналу високого рівня (HIGH), яка визначає біти «0» або «1».

Очікування завершення сигналу високого рівня:

```
loopCnt = 10000;
while(digitalRead(pin) == HIGH)
    if (loopCnt-- == 0) return DHTLIB_ERROR_TIMEOUT;
```

Знову встановлюється лічильник `loopCnt` для запобігання зацикленню.

`while (digitalRead(pin) == HIGH)` – чекає, поки сигнал на піні знову стане низьким (LOW).

Визначення значення біта

```
if ((micros() - t) > 40) bits[idx] |= (1 << cnt);
```

`micros() - t` – обчислює тривалість сигналу високого рівня. Якщо тривалість більше 40 мікросекунд, це означає, що переданий біт – «1». Інакше – «0».

`bits[idx] |= (1 << cnt)` – використовується побітова операція OR для встановлення відповідного біта в масиві bits. `(1 << cnt)` створює маску, яка зсуває «1» на позицію cnt.

Перехід до наступного байта:

```
  if (cnt == 0)
    {
        cnt = 7;
        idx++;
    }
    else cnt--;
}
```

`if (cnt == 0)` – якщо поточний байт заповнено (всі 8 бітів), оновлюється індекс (idx) для переходу до наступного байта.

`cnt = 7` – лічильник біта cnt скидається на 7 (найстарший біт наступного байта).

`else cnt--` – в іншому випадку лічильник зменшується для переходу до молодшого біта поточного байта.

Розподіл даних:

```
humidity    = bits[0];
temperature = bits[2];
```

Вологість і температура зчитуються з перших двох байтів відповідно.

Перевірка контрольної суми:

```
uint8_t sum = bits[0] + bits[2];

if (bits[4] != sum) return DHTLIB_ERROR_CHECKSUM;
return DHTLIB_OK;
```

Контрольна сума обчислюється як сума значень першого та третього байта. Якщо значення не співпадають з п'ятим байтом, повертається помилка.

Таким чином, розглянутий файл бібліотеки є реалізацією базового алгоритму роботи з DHT11, що включає:

– ініціацію;

– зчитування;

– обробку даних.

Файл бібліотеки є ключовою частиною роботи з датчиком DHT11, оскільки саме він інтерпретує сигнали датчика в цифрові дані, придатні для обробки. Також бібліотека забезпечує обробку типових помилок, таких як таймаут та невідповідність контрольної суми.

3.2 Формування виконавчого скетчу

Сформуємо скетч для виконання та організації вимірювань температури та вологості.

Пояснимо по рядках написання скетчу. Почнемо з підключення бібліотеки та визначення параметрів:

```
#include "DHT.h"
```

Бібліотека «DHT.h» надає інструменти для роботи з датчиками серії DHT, такими як DHT11 і DHT22. Вона включає методи для ініціалізації датчика, зчитування температури, вологості та перевірки стану.

```
#define DHTPIN 4
```

Макрос DHTPIN визначає номер піна Arduino, до якого підключений датчик. У нашому випадку, датчик підключений до цифрового піна 4.

Вибір типу датчика

```
#define SENSOR_TYPE DHT22
// #define SENSOR_TYPE DHT11
```

Макрос SENSOR_TYPE задає тип датчика. Ми обираємо між моделями DHT22 та DHT11, змінивши значення:

– для DHT22 – залишимо `#define SENSOR_TYPE DHT22;`

– для DHT11 – розкоментуємо рядок `#define SENSOR_TYPE DHT11.`

Ініціалізація датчика:

```
DHT dht(DHTPIN, SENSOR_TYPE);
```

Об’єкт `dht` створюється з використанням двох параметрів:

– DHTPIN – пін, до якого підключений датчик;

– SENSOR_TYPE – тип датчика (DHT22 або DHT11), визначений через макрос.

Це дозволяє налаштовувати датчик автоматично, залежно від вибраного типу.

Функція `setup()`:

```
void setup() {
  Serial.begin(9600);  // Ініціалізація серійного порту
  dht.begin();         // Ініціалізація датчика
}
```

– `Serial.begin(9600)`: ініціалізує серійний порт для обміну даними зі швидкістю 9600 бод. Це потрібно для виводу інформації в «Монітор порту».

– `dht.begin()`: запускає внутрішні процеси датчика, готуючи його до роботи.

Основний цикл `loop()`:

```
void loop() {
  delay(2000);
```

– `delay(2000)`: встановлює затримку у 2 секунди між вимірюваннями. Це необхідно, оскільки датчики серії DHT не можуть проводити вимірювання частіше ніж один раз на 1 – 2 секунди.

Зчитування вологості:

```
float h = dht.readHumidity();
```

`dht.readHumidity()`: метод зчитує відносну вологість у відсотках (%). Результат записується у змінну «h» типу float.

Зчитування температури:

```
float t = dht.readTemperature();
```

– `dht.readTemperature()`: метод зчитує температуру в градусах Цельсія (°C). Результат зберігається у змінній t типу float.

Перевірка коректності даних:

```
if (isnan(h) || isnan(t)) {
  Serial.println("Не вдалося зчитати покази");
  return;
}
```

– `isnan(h)` та `isnan(t)` – перевіряють, чи є зчитані значення NaN (Not a Number). Це може статися через несправність датчика або проблеми з підключенням.

Якщо дані некоректні, виводиться повідомлення:

```
Не вдалося зчитати покази
```

Далі функція `loop` завершує свою ітерацію за допомогою `return`.

Виведення даних у серійний порт:

```
Serial.print("Вологість: ");
Serial.print(h);
Serial.print(" %\tТемпература: ");
Serial.print(t);
Serial.println(" *C");
```

– `Serial.print()`: використовується для послідовного виводу тексту та значень у монітор порту.

Виводиться інформація у форматі (приклад):

```
Вологість: 45.5 %  Температура: 22.8 *C
```

3.4 Особливості створеного коду та можливості його вдосконалення

Таким чином, після формування скетчу для вимірювання рівня вологості та температури, зосередимося на його характерних особливостях та розглянемо можливості щодо покращення скетчу.

Особливості:

а) автоматичний вибір датчика – легко перемикатися між моделями DHT11 і DHT22, змінюючи лише макрос SENSOR_TYPE;

б) перевірка даних – код обробляє помилки зчитування, що зменшує ризик отримання некоректних результатів;

в) простота налаштування – користувачеві достатньо вказати номер піна та тип датчика, решта налаштувань виконується автоматично.

Можливі подальшого вдосконалення:

а) додавання виводу у Fahrenheit – для цього слід використати метод `dht.readTemperature(true)`, який повертає температуру у Фаренгейтах;

б) використання OLED або LCD-дисплея – можна організувати підключення дисплею для локального відображення даних;

в) передача даних через Wi-Fi – можна долучити та використати модуль ESP8266 або ESP32 для організації віддаленого моніторингу.

4 РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ

Виконаємо складання схеми, як було описано у підрозд. 2.2 та підключимо спеціалізоване програмне забезпечення, як це було описано у підрозд. 3.1, а також завантажимо створений у підрозд. 3.2 скетч (рис. 4.1).

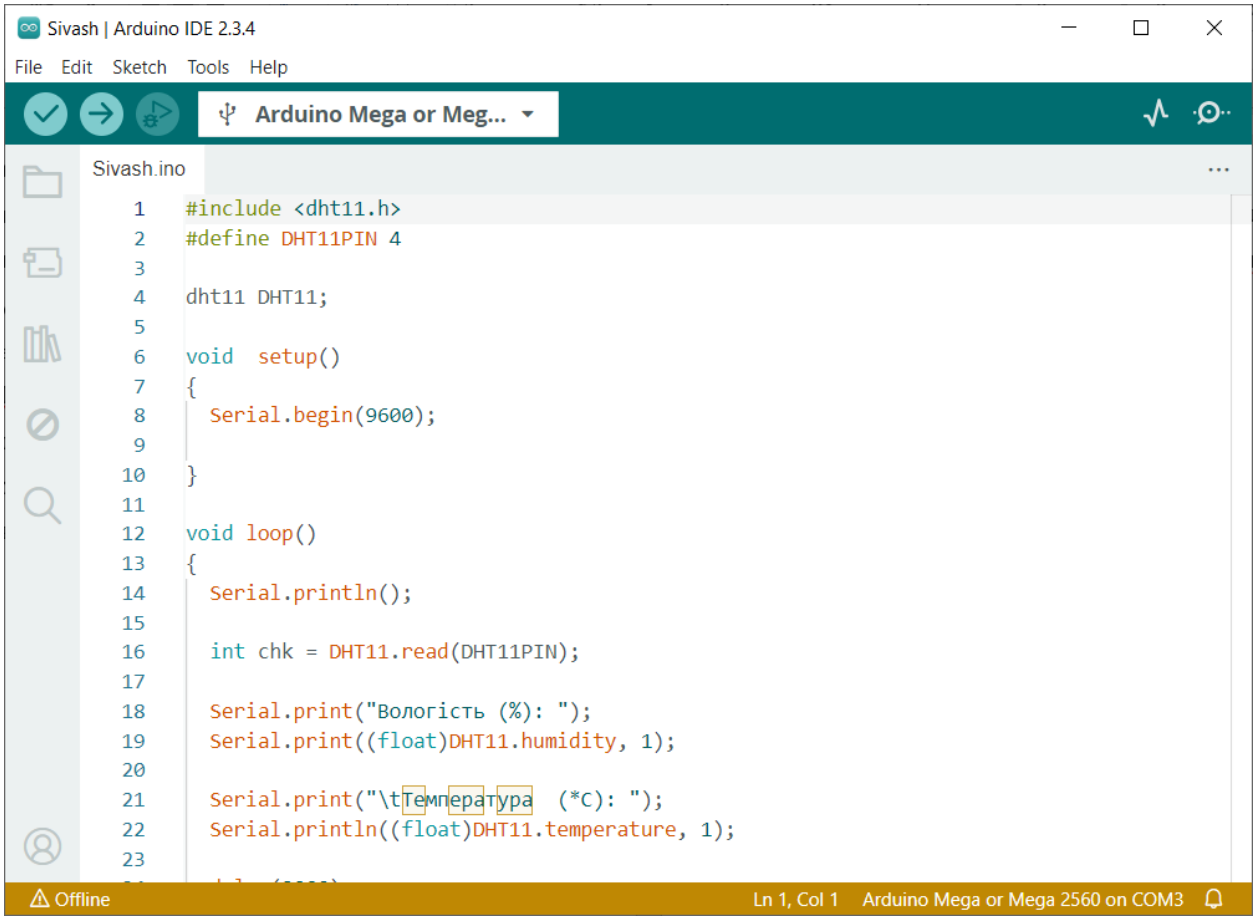

Рисунок 4.1 – Вигляд коду розробленого скетчу у вікні редактора

Виконаємо верифікацію створеного скетчу та отримаємо вихідне системне повідомлення (рис. 4.2).

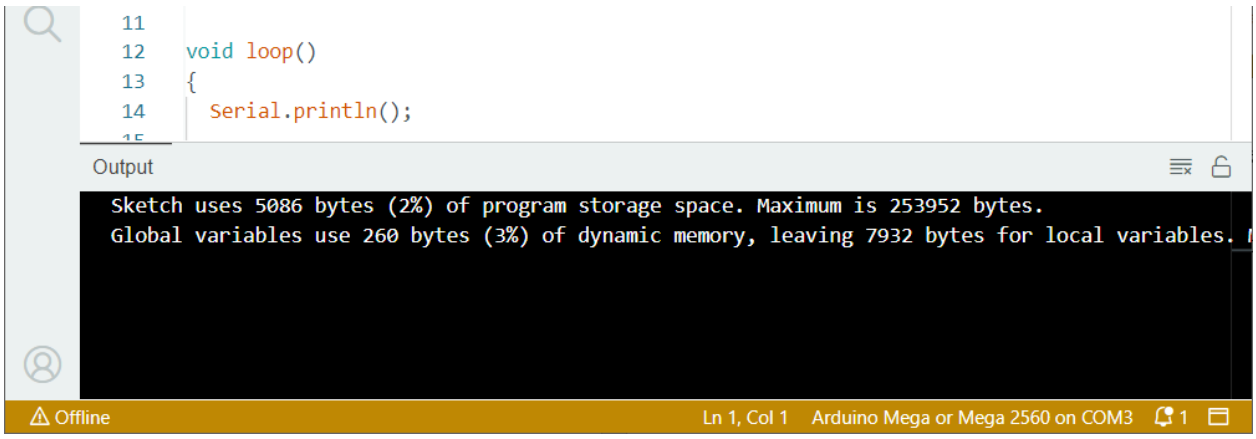

Рисунок 4.2 – Верифікація скетчу

Завантажимо скетч на виконання (функція «Upload») та відкриємо «Монітор порту» (рис. 4.3).

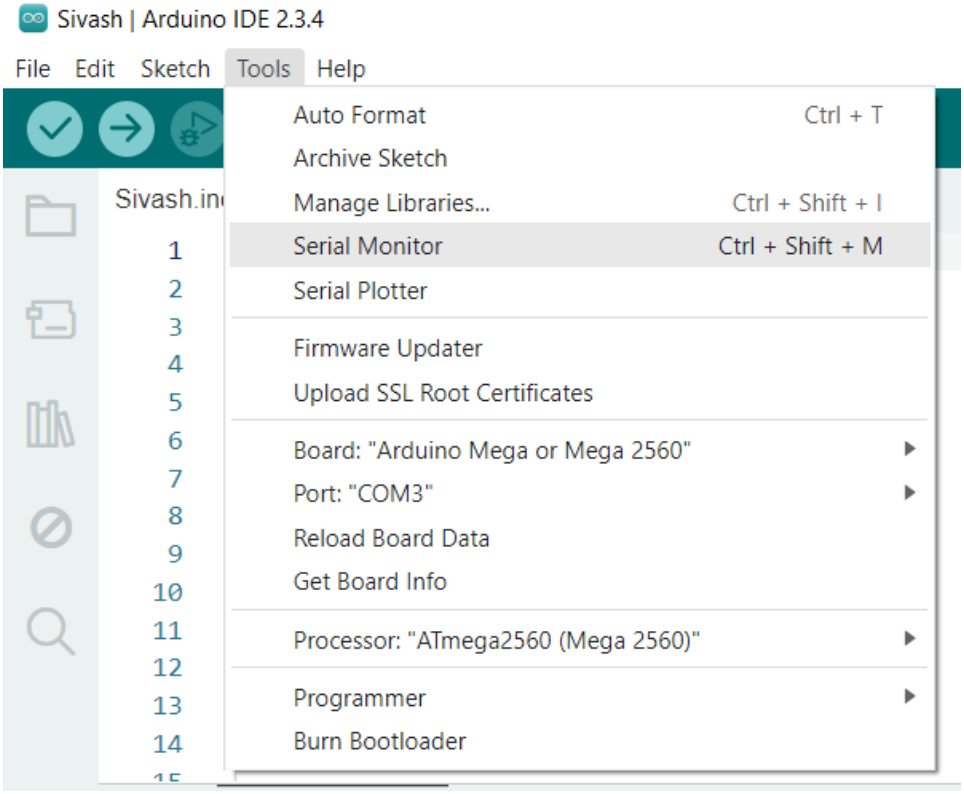

Рисунок 4.3 – Активація повідомлень «Монітору порту»

Монітор послідовного порту починає відображати передану інформацію щодо поточних значень вологості та температури кожні 2 с (2000 мс) як це показано на рис. 4.4.

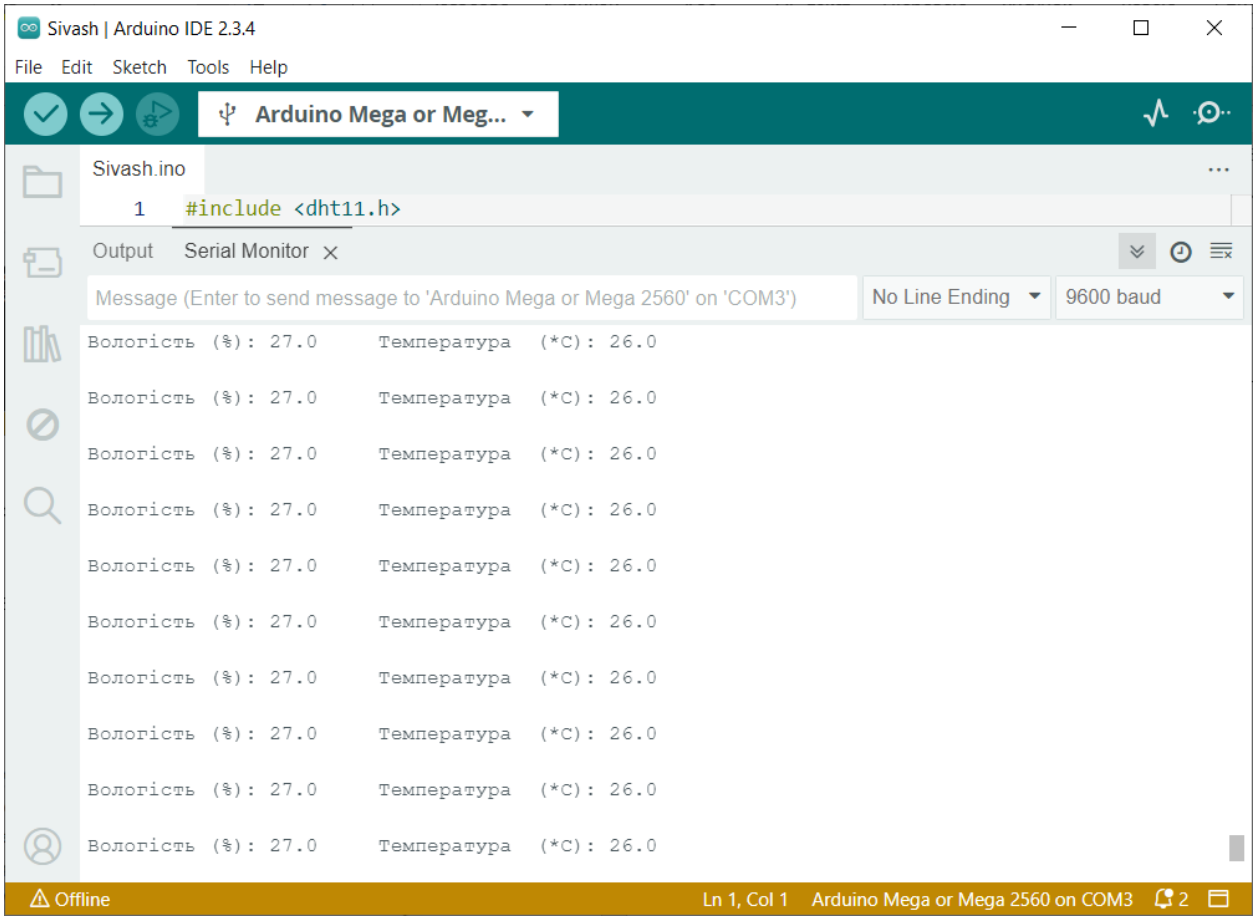

Рисунок 4.4 – Монітор послідовного порту із інформацію щодо значень вологості та температури

Таким чином, вимірювання, обробка та передача даних вологості та температури на базі Arduino є не лише корисним практичним пристроєм, а й прикладом ефективного використання сучасних мікроконтролерних технологій для вирішення актуальних задач у сфері моніторингу погодних умов.

Успішна реалізація даного проєкту може дозволити виконувати агрометеорологічні вимірювання та сприятиме подальшому розвитку автоматизованих систем моніторингу навколишнього середовища.

ВИСНОВКИ

У ході аналізу та реалізації роботи з мікроконтролерами Arduino та сенсорами, зокрема температури і вологості DHT11, було виконано значний обсяг досліджень і практичних завдань, що дозволяють зробити наступні висновки.

1 Проаналізовано особливості платформи Arduino:

а) Arduino – це зручна й універсальна платформа для розробки електронних пристроїв, яка має широкі можливості завдяки апаратному та програмному забезпеченню;

б) використання мікроконтролера Arduino дає змогу легко підключати сенсори та виконавчі пристрої завдяки наявності стандартних цифрових і аналогових входів / виходів;

в) Arduino IDE забезпечує простий інтерфейс для програмування, що робить платформу доступною навіть для новачків.

2 Складено схему підключення сенсора DHT11:

а) DHT11 – це цифровий датчик температури та вологості, який має простий інтерфейс передачі даних через одну сигнальну лінію;

б) у схемі використовується підтягувальний резистор, що забезпечує стабільність сигналу на лінії передачі даних.

в) правильне підключення сенсора включає використання окремих ліній для живлення (+5В), землі (GND) та сигнальної лінії, підключеної до цифрового входу мікроконтролера.

3 Розроблено програмне забезпечення для роботи з DHT11:

а) програмна реалізація базується на бібліотеці DHT, яка спрощує процес отримання даних із сенсора;

б) у програмному коді використовуються функції для:

– ініціалізації сенсора;

– зчитування температури;

– зчитування вологості;

– перевірки коректності отриманих даних.

в) використання перевірки на наявність помилок (функція isnan()) дозволяє виявляти збої в роботі сенсора або зчитуванні даних, забезпечуючи надійність системи.

4 Виконано аналіз роботи алгоритму читання даних:

а) процес читання даних із сенсора базується на зчитуванні 40 бітів інформації, що включає значення:

– вологості;

– температури;

– контрольної суми;

б) реалізація алгоритму передбачає перевірку на таймаути під час зчитування кожного біту, що дозволяє уникати зависання програми при несправності сенсора;

в) завдяки аналізу кожного етапу обміну даними, було детально вивчено механізми передачі цифрової інформації між сенсором та Arduino.

5 Розширено функціональності метеостанції:

а) додано можливість вибору сенсора (DHT11 або DHT22) у програмному коді, що забезпечує гнучкість системи;

б) схема підключення та програмна частина можуть бути легко модифіковані для роботи з іншими датчиками або додатковими пристроями.

Практична значущість проекту:

– реалізована система збору, обробки та передачі даних температури і вологості може бути застосована у різних сферах: від побутових метеостанцій до промислових систем моніторингу мікроклімату;

– простота реалізації та доступність компонентів дозволяють використовувати систему як навчальний проєкт для вивчення основ роботи з мікроконтролерами та сенсорами.

Рекомендації для подальших досліджень:

– розширення системи шляхом інтеграції з іншими сенсорами (CO2, освітленості тощо) для створення багатофункціональної метеостанції;

– використання бездротових модулів (наприклад, Wi-Fi або Bluetooth) для передачі даних на віддалений сервер або мобільний додаток;

– оптимізація коду для зменшення затримок і підвищення ефективності роботи системи в умовах багатозадачності.

Таким чином, проєкт, який пов’язаний із підключенням сенсора DHT11 до Arduino, дозволив дослідити:

– основи роботи мікроконтролерів;

– обмін даними через цифрові інтерфейси;

– реалізацію алгоритмів обробки сигналів.

Завдяки простоті реалізації та можливостям розширення, система має високий потенціал для практичного застосування та подальшого вдосконалення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 Офіційний сайт Arduino. URL: https://www.arduino.cc/ (дата звернення: 18.04.2025 р.).

2 Система довідки по компонентам Arduino. URL: https://docs.arduino.cc/hardware/ (дата звернення: 19.04.2025 р.).

3 База проєктів та інструкцій для Arduino. URL: https://www.instructables.com/ARDUINO-Tutorials/ (дата звернення: 20.04.2025 р.)

4 Гурток «Основи робототехніки на основі ARDUINO». URL: https://mcnttumzt.wordpress.com/fundamentals-of-robotics-based-on-arduino-group/ (дата звернення: 21.04.2025 р.).

5 DHT11 vs DHT22 vs LM35 vs DS18B20 vs BME280 vs BMP180. URL: https://randomnerdtutorials.com/dht11-vs-dht22-vs-lm35-vs-ds18b20-vs-bme280-vs-bmp180/ (дата звернення: 23.04.2025 р.).

6 Arduino IDE 2.3.4. URL: https://www.arduino.cc/en/software?utm_source=chatgpt.com (дата звернення: 25.04.2025 р.).

7 Using the Arduino Software (IDE). URL: https://docs.arduino.cc/learn/starting-guide/the-arduino-software-ide/ (дата звернення: 28.04.2025 р.).

8 Набір для побудови метеостанції на ESP8266 IOT від Elecrow. URL: https://arduino.ua/prod2599-nabor-dlya-postroeniya-meteostancii-na-esp8266-iot?srsltid=AfmBOorKvVGL0Des7ANS7gle92NS35bxY_zLz5YCFCLqsqmjkYMJNZ-c (дата звернення: 29.04.2025 р.).

9 Форум любителів електронних саморобок Arduino. URL: https://forum.arduinka.biz.ua/viewtopic.php?t=25&sid=3e8a6036ec6a2eb3cf0e7231de1b8382 (дата звернення: 01.05.2025 р.).

10 Метеостанція на Arduino з датчиком DHT-11. URL: https://stemua.science/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%D0%BA%D0%B8/%D0%BC%D0%B5%D1%82%D0%B5%D0%BE%D1%81%D1%82%D0%B0%D0%BD%D1%86%D1%96%D1%8F-

%D0%BD%D0%B0-arduino-%D0%B7-%D0%B4%D0%B0%D1%82%D1%87%D0%B8%D0%BA%D0%BE%D0%BC-dht-11/ (дата звернення: 02.05.2025 р.).

11 Колекція проєктів Arduino з покроковими інструкціями. URL: https://projecthub.arduino.cc/ (дата звернення: 04.05.2025 р.).

12 Датчики вологості для розумного будинку. URL: https://rozetka.com.ua/ua/umnie-datchiki/c4638399/tip-detektirovaniya=vlagnost/ (дата звернення: 05.05.2025 р.).