МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»

Освітньо-професійна програма: «Обслуговування комп'ютерних систем і мереж»

Група: 4КС-58

Дипломний проект

здобувача освіти денної форми навчання
КС.58.12.000.ДП

КЛІНОВСЬКОГО
ВОЛОДИМИРА ВІТАЛІЙОВИЧА

м. Одеса
2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **123 «Комп'ютерна інженерія»**

Освітньо-професійна програма: «**Обслуговування комп'ютерних систем і мереж»**

Група: **4КС-58**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

Розробка пристрою діагностики стану ПК на платформі Arduino

Проектний матеріал складається з пояснювальної записки на 92 сторінках та графічного (презентаційного) матеріалу на 16 аркушах (слайдах)

Дипломник _____ (Кліновський В.В.)

Керівник _____ (Кривченко А.А.)

Консультанти:

з економічного розділу _____ (Канський М.Ю.)

з розділу охорони праці та техніки безпеки _____ (Чорновол Н.І.)

з нормоконтролю _____ (Петрашова В.І.)

старший консультант _____ (Кривченко Ю.В.)

До захисту допущений

Голова циклової комісії _____ (Кривченко Ю.В.)

Завідувач відділення _____ (Краснокутська К.Г.)

Захист «20» червня 2025 р. Протокол ЕК № 1

Оцінка ЕК 5 (відмінно) / 90 з.

Секретар ЕК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення *комп'ютерних систем* Комісія *КТ та ПІ*

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Обслуговування комп'ютерних систем і мереж»*

ЗАТВЕРДЖУЮ:

Заст. дир. з НВР

Беркань І.В.

"19" 05 2025 р.

ЗАВДАННЯ

на дипломний проект

здобувачеві (здобувачці) освіти *Кліновському Володимиру Віталійовичу*

(прізвище, ім'я, по батькові)

1. Тема проекту *Розробка пристрою діагностики стану ПК на платформі Arduino*

затверджена наказом по коледжу від "14" 11 2024 р.

№ 246

2. Термін здачі закінченого проекту

16.05.25

3. Вихідні данні до проекту *1. У якості мікроконтролеру для обчислювальної платформи використовувати Arduino Nano. 2. Візуалізувати ключові параметри ПК у режимі реального часу. 3. Забезпечити відображення вимірюваних даних із ПК на OLED-дисплей; 4. Розробити пристрій виводу даних з ПК.*

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

Аналіз застосування приладів для контролю показників системи ПК. Визначення основних показників системи ПК. Визначення технічних характеристик пристрою. Розробка моделі пристрою і визначення базових параметрів. Підбір елементів пристрою для контролю показників системи ПК та їх аналіз. Створення схеми та інтеграція компонентів пристрою для контролю показників системи ПК. Реалізація програмного забезпечення мікроконтролера. Аналіз результатів роботи пристрою контролю показників системи ПК.

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

Вибір елементної бази: мікроконтролери; вибір елементної бази: дисплеї; апаратна частина пристрою; дослідження програмних засобів; загальна архітектура системи; схема підключення OLED-дисплея до Arduino Nano; узагальнена блок-схема; фрагменти коду прошивки: підключення бібліотек та ініціалізація дисплея алгоритму вимірювань; фрагменти коду Python-скрипт для передачі даних на Arduino; фрагменти коду для Arduino Nano; Фото зібраного пристрою після тестової компоновки; фото монтування корпусу; підключений до ПК пристрій; схема тестування приладу імітація зміни стану ПК.

6. Консультанти по проекту, із зазначенням розділів проекту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основний розділ	*Кривченко А.А.*		
Економічний розділ	*Канський М.Ю.*		
Розділ охорони праці	*Чорновол Н.І.*		
Нормоконтроль	*Петрашова В.І.*		
Старший консультант	*Кривченко Ю.В.*		

7. Дата видачі завдання 02.06.25

Керівник *Кривченко А.А.* _____ (підпис)

Завдання прийняв до виконання _____ (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проекту	Термін виконання етапів дипломного проекту	Відмітка про виконання
1.	*Вступ. Постановка задачі проектування*	22.05.25	Виконав
2.	*Аналіз технічного завдання та загальна концепція*	23.05.25	Виконав
3.	*Аналіз комерційних рішень для виведення системної*	25.05.25	Виконав
	інформації.		
4.	*Аналіз ключових параметрів діагностики ПК*	27.05.25	Виконав
5.	*Аналіз характеристик існуючих приладів для*	28.05.25	Виконав
	контролю показників системи ПК.		
6.	*Розробка структури пристрою і розрахунок*	30.05.25	Виконав
	параметрів		
7.	*Вибір елементної бази та аналіз компонентів*	31.05.25	Виконав
8.	*Розробка схеми підключення компонентів пристрою*	01.06.25	Виконав
9.	*Розробка програмного забезпечення для*	02.06.25	Виконав
	мікроконтролера Arduino Nano		
10.	*Збірка пристрою*	03.06.25	Виконав
11.	*Тестування передачі даних вимірювань*	04.06.25	Виконав
12.	*Виконання економічних розрахунків*	05.06.25	Виконав
13.	*Розробка заходів з охорони праці та техніки безпеки*	07.06.25	Виконав
14.	*Виконання графічної частини проекту*	09.06.25	Виконав
15.	*Підготовка проекту до захисту та складання*	12.06.25	Виконав
	доповіді		

Дипломник _____ (підпис)

Керівник _____ (підпис)

Форма	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			КС 58. 12 000. 00 ДП	Дипломний проект		
А4			КС 58. 12 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	КС 58. 12 000. 00 ДП		
Розробив		Кліновський В.В.		16.06.25	Розробка пристрою діагностики стану ПК на платформі Arduino	Літ.	Аркуш 4 / Аркушів 92
Перевірив		Кривченко А.А.		16.06.25		Н Д П	
Н. Контр.		Петрашова В.І.		16.06.25		ВСП "ОТФК ОНТУ" гр.4КС-58	
Затверд.		Кривченко Ю.В.		16.06.25			

ЗМІСТ

ВСТУП

В сучасному світі складно уявити повсякденне життя та професійну діяльність без персонального комп'ютера. Вони використовуються всюди: офісна робота, професійна обробка фото та відео, чи відпочинок у вільний час, проведений як за улюбленими іграми, так і перегляд фільмів, — та багато інших аспектів життя. Надмірне навантаження на апаратне забезпечення, перегрів або нестабільна робота елементів можуть призводити до зниження продуктивності, втрати даних або навіть фізичного виходу з ладу обладнання. Постійний контроль апаратних параметрів допомагає підтримувати оптимальну продуктивність комп'ютера та запобігати збоям.

Існуючі програмні засоби моніторингу, такі як «Диспетчер завдань» у Windows, HWMonitor, AIDA64, чи спеціалізовані утиліти для графічних карт (наприклад, MSI Afterburner для NVIDIA та AMD), надають вичерпну інформацію про стан компонентів ПК, включаючи температурні показники, завантаження окремих ядер процесора, використання відеопам'яті та зміни робочих частот. Але користувачам часто потрібен автономний і завжди доступний індикатор стану системи, який би надавав критичну інформацію без необхідності переключатися між вікнами програм. Тому розробка мобільного пристрою, сумісного з актуальними представниками сімейства ОС Windows, який забезпечує отримання, обробку та візуалізацію ключових параметрів персонального комп'ютера в режимі реального часу, є актуальною та перспективною задачею.

У представленному проєкті відстежуються три ключові параметри апаратного моніторингу стану ПК: навантаження CPU (%) — показує рівень використання центрального процесора, що впливає на швидкодію системи; навантаження GPU (%) — важливий показник, який визначає продуктивність графічного процесора; використання RAM (%) — контроль оперативної пам'яті, так як її перевантаження може уповільнювати роботу ПК.

Обрано саме ці параметри, щоб отримати максимально повну картину про «самопочуття» ПК для широкого кола користувачів, відстежуючи потенційні «вузькі місця» продуктивності.

1 ОСНОВНИЙ РОЗДІЛ

1.1 Аналіз технічного завдання та визначення засобів реалізації

Ця розробка має на меті створення бюджетного, портативного пристрою, який забезпечує апаратний моніторинг стану ПК та інформує користувача про рівень завантаження основних компонентів комп'ютера.

Параметри відстеження стану відібрані тому, що допомагають оцінити продуктивність комп'ютера та вчасно виявити потенційні проблеми, а саме:

Навантаження CPU (%): відсоток використання центрального процесора. Високе значення вказує на інтенсивну обробку даних, що може бути причиною уповільнення системи чи сигналом про наявність ресурсоємних процесів.

Навантаження GPU (%): відсоток використання графічного процесора. Важливо: для геймерів та професіоналів, що працюють з графікою, оскільки відображає здатність системи рендерити зображення чи відео.

Використання RAM (%): відсоток зайнятої оперативної пам'яті. Надмірне використання ОЗП може призвести до використання файлу підкачки, що значно уповільнює систему.

Основні вимоги до пристрою: зчитування параметрів ПК; передача даних через COM-порт від ПК до мікроконтролера; виведення інформації на OLED-дисплей, що забезпечує високу контрастність і чітке зображення.

Для проєктуваня пристрою планується використати наступні компоненти: мікроконтроллер, OLED-дисплей SSD1306 0.96, Кабелі Dupont (Мати-Мати).

Також, планується використати наступне програмне забезпечення: Open Hardware Monitor (далі OHM), який надає дані у форматі JSON через веб-інтерфейс. Ці дані будуть оброблятись Python-скриптом: виконає HTTP-запит до Open Hardware Monitor, парсить отриманий JSON, виділяючи ключові параметри (CPU, GPU, RAM), сформує текстову строку та передає її через Serial на Arduino Nano.

Після збору необхідних компонентів буде почато створення пристрою, а саме: побудована схема підключення; підключення OLED-дисплея з Arduino Nano через I2C; підключення Arduino Nano до ПК; написання Python-скрипта для збору

даних; програмування мікроконтролера; перевірка роботи пристрою; фінальні налаштування.

У результаті виконання усіх цих дій буде отримано пристрій, який буде відображати заплановані параметри ПК, а легка інтеграція та простота експлуатації дозволить використовувати пристрій без спеціальних знань.

1.2 Огляд комерційних рішень для виведення системної інформації

Зовнішні пристрої моніторингу, які виводять показники системи на окремі дисплеї набувають все більшої популярності. Такі рішення можуть використовуватися для:

– Серверів — де важливо безперервно контролювати завантаження CPU чи температуру системи;

– Геймерських ПК — для зручного виведення FPS, температури, завантаження компонентів;

– Інженерних систем — де потрібен технічний контроль за станом пристроїв;

– Майстерень з обслуговування комп’ютерів — для демонстрації стану обладнання без додаткового програмного забезпечення (далі ПЗ).

Ось кілька реальних пристроїв, які виконують подібні функції моніторингу стану ПК:

3.4inch USB Monitor W — це пристрій для ПК невеликий LCD-/кран діагоналлю 3.4 дюйми, який підключається через USB і призначений для встановлення всередині комп'ютерного корпусу. Його основне призначення — відображення важливої інформації про систему, такої як температура процесора, використання оперативної пам'яті, швидкість обертання вентиляторів, та інші параметри, що стосуються продуктивності ПК.

Динамічний відеофон Вбудований високопродуктивний процесор для створення інтерфейсу користувача та фонового відео. Для живлення та передачі даних потрібен лише один кабель USB, який легко підключити без складних операцій.

Програмне забезпечення власної розробки одним кліком миші Просте використання без складних операцій, зрозумілий інтерфейс користувача.

Жодного впливу на продуктивність та роботу комп'ютера Не зменшує продуктивність комп'ютера та майже не використовує графічний процесор з наданим нами програмним забезпеченням. Екран буде розпізнано як USB-пристрій, тому миша не переміститься на цей екран.

Моніторинг різних даних у реальному часі ЦП/ Графічна карта/ Пам'ять/ Жорсткий диск/ Мережа/ інше (модель плати/ місто/ погода/ дата, час/ гучність)

Надійність корпусу Корпус із алюмінієвого сплаву, жорсткий і міцний, що покращує загальну якість.

Рисунок 1.1. Приклад пристрою 3.4inch USB Monitor W

7" USB IPS монітор "другий екран" 1024×600 від Waveshare — 7"USB монітор, який можна використовувати як додатковий екран корпусу ПК. Панель оснащена IPS матрицею з роздільною здатністю 800×480 З функцією аналізу музичного спектру, що дозволяє використовувати дисплей для візуалізації звуку у реальному часі. Монітор виконаний в металевому корпусі з USB Type-C роз'ємом, Для максимального використання можливостей екрану написана програма управління для ПК. Для максимального використання потенціалу дисплея виробник надає фірмове програмне забезпечення для керування контентом, відображення системної інформації та гнучкого налаштування графічних елементів.

Рисунок 1.2. Приклад пристрою 7" USB IPS монітор "другий екран"

NZXT Kraken Elite RGB 280 мм AIO liquid cooler w/Display, RGB Fans White – хоча основне призначення цього пристрою це система рідинного охолодження вона також оснащена вбудованим IPS LCD IPS LCD-дисплеєм діагоналлю 2.72" з роздільною здатністю 640×640 пікселів та частотою оновлення 60 Гц. для виведення показників ПК. Завдяки цьому на нього можна виводити не тільки показники з ПК, а також різноманітні фото відео та музичні плеєри.

Рисунок 1.3. Приклад пристрою NZXT Kraken Elite

У проєкті, що розроблюється, на базі Arduino Nano та OLED-дисплея SSD1306 вигідно відрізняється насамперед своєю винятковою економічністю та

гнучкістю. Тоді як комерційні рішення є дорожчими та мають обмежені можливості кастомізації, цей пристрій дозволяє повністю контролювати виведення інформації, її формат та візуалізацію. Завдяки відкритому програмному коду та використанню популярних бібліотек, можливо адаптувати функціонал під будь-які потреби, додаючи нові показники чи змінюючи дизайн. Це робить пристрій ідеальним для ентузіастів, студентів та тих, хто цінує можливість самостійного налаштування та розширення функціоналу, пропонуючи ефективне рішення за мінімальну вартість.

Порівняльний аналіз доводить доцільність створення автономного апаратного пристрою для моніторингу стану ПК як з технічної, так і з користувацької точок зору.

1.3 Аналіз ключових параметрів діагностики ПК

Щоб створити ефективний пристрій діагностики потрібно чітко визначити основні параметри за якими буде проводитись відстежування стану ПК. Суттєві зміни цих параметрів можуть вказувати на перевантаження і, навіть, несправності у роботі системи ПК.

Основні діагностичні параметри ПК:

– Температура центрального процесора (CPU Temperature);

Висока температура може свідчити про проблеми з охолодженням чи перевантаження системи.

Нормальний діапазон: 30–70 °C, критичний — вище 85 °C.

– Завантаження центрального процесора (CPU Load);

Відображає ступінь зайнятості процесора обчислювальними задачами. Саме цей показник дозволяє користувачеві оцінити, чи не перевантажений комп’ютер фоновими процесами, та вчасно виявити потенційні проблеми з продуктивністю.

Нормальний діапазон: до 60% у стані простою.

– Температура графічного процесора (GPU Temperature);

Особливо важлива при роботі з графікою, 3D-редагуванням та в іграх.

Нормальний діапазон: 40–80 °C.

– Завантаження графічного процесора (GPU Load);

Дозволяє оцінити інтенсивність використання відеокарти.

– Використання оперативної пам’яті (RAM Load);

Надмірне використання RAM часто призводить до уповільнення системи через звернення до файлу підкачки на жорсткому диску, SSD. Моніторинг RAM дозволяє оперативно зреагувати на ситуації, коли об’єм оперативної пам’яті є вузьким місцем у продуктивності ПК.

Оптимально: не більше 75% при звичайному використанні.

– Температура материнської плати та чипсету

Менш критичний, але також важливий параметр для діагностики внутрішнього перегріву.

Температура жорсткого диска (HDD/SSD Temp)

Перегрів накопичувача може призвести до втрати даних.

Рекомендовано: до 50 °C.

– Причини обрання саме цих параметрів:

– Інформативність — дані прямо вказують на навантаження основних компонентів системи, перегрів процесора або відеокарти, а також збої в роботі апаратного забезпечення;

– Доступність через Open Hardware Monitor API;

– Легкість парсингу та обробки Python-скриптом;

– Актуальність для будь-якого типу ПК — офісного, ігрового, серверного.

Параметри, які виводяться на зовнішній дисплей у данній реалізації:

– Завантаження CPU (у %);

– Завантаження GPU (у %);

– Завантаження RAM (у %).

Ці дані обираються як базові для індикації, так як є найбільш динамічними, швидко реагують на зміни у роботі ПК та наочно демонструють навантаження системи і дозволяють ефективно відстежувати її стан у реальному часі.

1.4 Вибір елементної бази

Перед початком створення проєкту однією із ключових цілей стало обрання оптимальної апаратної платформи, яка буде відповідати всім технічним вимогам проєкту. У цьому розділі буде виконано ґрунтовне дослідження та порівняльний аналіз кількох популярних моделей мікроконтролерів, а саме: Arduino Nano, Arduino Uno, ESP32, Raspberry Pi Pico. А також обґрунтування вибору найбільш відповідної платформи для забезпечення стабільної роботи пристрою при мінімальних витратах та максимальній простоті реалізації.

Перед тим, як порівнювати платформи, визначаємо ключові критерії, які будуть враховувати при виборі:

– фізичні розміри та компактність: пристрій має бути портативним і компактним для зручності розміщення біля комп'ютера чи, у випадку необхідності, навіть вбудовуватися у корпус ПК;

– кількість входів/виходів (I/O): достатня для підключення OLED-дисплея, потенційних сенсорів чи індикаторів;

– наявність інтерфейсів I^2C/UART: для зв'язку з дисплеєм та ПК;

– простота програмування: наявність бібліотек, інструментів розробки, активної спільноти;

– енергоспоживання: низьке споживання є бажаним;

– Вартість та доступність: важливий фактор при обмеженому бюджеті чи масовому виробництві.

Arduino Uno

Одна з найвідоміших платформ на базі мікроконтролера ATmega328P. Вона має 14 цифрових входів/виходів, з яких 6 можуть бути використані як ШІМ-виходи, та 6 аналогових входів. Uno має достатньо ресурсів для більшості базових проєктів.

Переваги: простота використання, широка підтримка бібліотек, хороша документація.

Недоліки: великі габарити, відсутність micro-USB (використовується

USB-B), вища вартість у порівнянні з Nano.

Arduino Uno має габарити приблизно 68.6 мм × 53.4 мм, що робить її малопридатною для проєктів, де розмір є критичним.

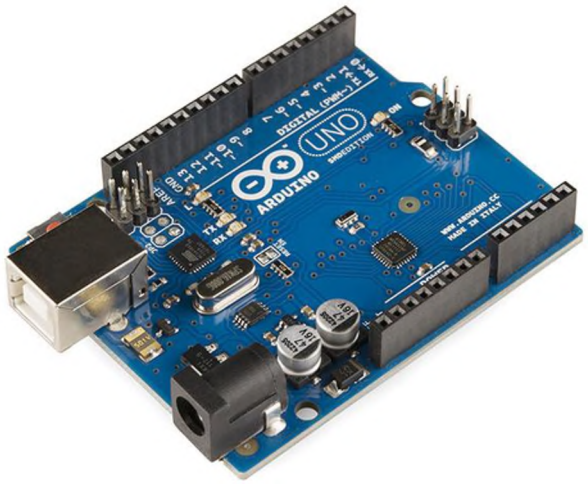


Рисунок 1.4. Приклад пристрою Arduino Uno

Arduino Nano

Є мініатюрною версією Uno, заснованою на тому ж мікроконтролері ATmega328P. Вона має всі основні функції Uno, але в більш компактному корпусі — 45 мм × 18 мм.

Переваги: Дуже компактний розмір, повна підтримка середовища Arduino IDE, Micro-USB інтерфейс для живлення та програмування, низьке енергоспоживання.

Недоліки: обмежена кількість цифрових та аналогових пінів порівняно з більшими платами (але цілком достатньо для даного проєкту).

Arduino Nano ідеально підходить для компактних вбудованих систем. Вона має всі необхідні інтерфейси: UART, I^2C, SPI, а також підтримує підключення OLED-дисплеїв через I^2C без використання додаткових модулів.

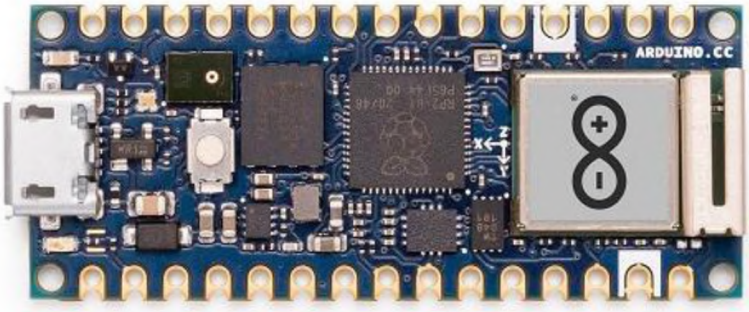


Рисунок 1.5. Приклад пристрою Arduino Nano

Raspberry Pi Pico

Плата на базі мікроконтролера RP2040 з двоядерним ARM Cortex-M0+. Вона має велику кількість цифрових пінів, підтримує C/C++ та MicroPython.

Переваги: потужна платформа для розробки, підтримка MicroPython, дешева вартість.

Недоліки: складніший запуск для новачків, обмежена підтримка деяких Arduino-бібліотек.

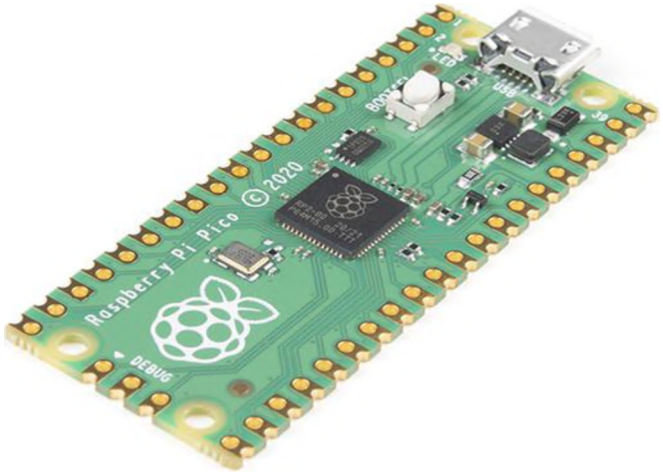


Рисунок 1.6. Приклад пристрою Raspberry Pi Pico

Для проєкту, що вимагає простоти й стабільності, Pico може бути надмірним і потребує адаптації бібліотек.

ESP32

Потужна 32-бітна мікроконтролерна платформа на базі архітектури Xtensa LX6 з двоядерним процесором, Wi-Fi та Bluetooth із частотою до 240 МГц. Вона надає значно більшу обчислювальну потужність та оперативної та флеш-пам’яті у порівнянні з Arduino.

Переваги: висока продуктивність, вбудовані WI-FI та Bluetooth, багато пінів, низьке енергоспоживання у режимах сну

Недоліки: складніше у налаштуванні, більш висока вартість, не всі OLED-дисплеї одразу сумісні.

Попри свою потужність, ESP32 у межах даного проєкту є надлишковим рішенням для проєкту, що не потребує підключення до Інтернету і складної обробки даних.

Рисунок 1.7. Приклад пристрою ESP32

У табл.1.1. здійснено порівняння апаратних платформ:

Таблиця 1.1. Порівняльний аналіз апаратних платформ

Параметр	Arduino Nano	Arduino Uno	ESP32	Raspberry Pi Pico
Кількість входів/виходів (I/O)	14 цифрових, 8 аналогових	14 цифрових, 6 аналогових	34 цифрових, 18 аналогових	26 цифрових, 3 аналогових
Наявність інтерфейсів I^2C/UART	Є, I^2C/UART	Є, I^2C /UART	Є, I^2C /UART	Є, I^2C /UART
Простота програмування	Простий, велика спільнота	Простий, велика спільнота	Складніший, потребує Wi-Fi налаштувань	Доступний, має підтримку MicroPython
Енергоспоживання	Низьке	Низьке	Вище середнього	Дуже низьке
Вартість та доступність	Середня, широко доступний	Середня, широко доступний	Висока	Дешева альтернатива

Після проведення аналізу, було обрано Arduino Nano тому що:

Компактність: пристрій не займає багато місця (розмір плати становить всього 45x18 мм) та легко може інтегруватись у корпус ПК чи на панель, і робить кінцевий пристрій портативним та лаконічним на робочому столі.

Наявність необхідних інтерфейсів: підтримка I^2C для OLED-дисплею, його можливостей цілком достатньо для виконання поставлених завдань. Для

зберігання програми (скетчу) він оснащений 32 КБ Flash-пам'яті, 2 КБ SRAM для оперативних даних та 1 КБ EEPROM для постійного зберігання даних. У проєкті, що включає в себе бібліотеки Adafruit GFX та Adafruit SSD1306 для керування OLED-дисплеєм, логіку серійного зв'язку та парсингу отриманих рядків, 32 КБ Flash-пам'яті є більш ніж достатнім обсягом. SRAM у 2 КБ також вистачить для зберігання тимчасових змінних, буферів для серійних даних та буфера кадру для дисплея. Тактова частота мікроконтролера 16 МГц забезпечує достатню швидкість обробки даних, отриманих по серійному порту, та їх швидке відображення на OLED-дисплеї без помітних затримок.

Швидкість передачі (baud rate) може бути налаштована, зазвичай використовуються значення 9600, 19200, 38400, 57600 та 115200 бод. Для передачі трьох невеликих числових значень (навантаження CPU, GPU, RAM) навіть 9600 бод буде достатньо, забезпечуючи надійну передачу даних без втрат.

Сумісність: Arduino IDE повністю підтримує Nano, є багато прикладів та бібліотек.

Надійність: перевірена часом архітектура, велика кількість навчальних матеріалів

Вартість: одна з найдешевших платформ при належному функціоналі.

Також Arduino Nano дозволяє легко реалізувати алгоритм обробки вхідних даних з послідовного порту, що надходять із Python-скрипта. Завдяки підтримці OLED-бібліотек, розробка інтерфейсу виводу даних була реалізована просто й швидко.

Також, у разі виникнення подальшої необхідності у розширенні функціоналу проєкту можуть бути використані наступні платформи та елементи такі як: перехід на більш потужні платформи, такі як ESP32 і Raspberry Pi Pico. Вони більш універсальні що дозволяє додати нові функції у проєкт, такі як: передавати дані по Bluetooth чи Wi-Fi, ESP32 стане логічною заміною. Однак для базової версії пристрою, яка виконує функції локального моніторингу та відображення, Nano є найбільш збалансованим рішенням.

Також, буде проведено дослідження та порівняльний аналіз кількох

популярних моделей дисплеїв, а саме: OLED 0.96" (SSD1306), TFT 1.8" (ST7735), LCD 16x2 (на контролері HD44780). А також обґрунтування вибору найбільш відповідного для забезпечення стабільної роботи пристрою при мінімальних витратах та максимальній простоті реалізації.

OLED 0.96" (SSD1306)

Цей дисплей має високу контрастність завдяки технології OLED, кожен піксель світиться самостійно. Підключається через I^2C, має невелику споживану потужність, але обмежену кількість кольорів.

Переваги: висока контрастність, чітке зображення навіть при малих розмірах; підключення лише по 2 дротах через I^2C, що економить піни Arduino.

Недоліки: монохромний — відображає лише білий чи синій колір (залежно від моделі); обмежений розмір екрану, що може бути недостатнім для великої кількості інформації.

Рисунок 1.8. Приклад OLED 0.96"

TFT 1.8" (ST7735)

Дисплей на основі рідкокристалічної технології з кольоровою передачею та високою роздільною здатністю.

Переваги: кольоровий дисплей з можливістю виводу графіки, зображень, значків; вища роздільна здатність порівняно з OLED.

Недоліки: складніше підключення (SPI, більше проводів); вище енергоспоживання, що може бути критично для живлення від USB чи батареї.

Рисунок 1.9. Приклад TFT 1.8"

LCD 16x2 (HD44780)

Класичний дисплей для Arduino. Відображає лише текст (2 рядки по 16 символів), не підтримує графіку.

Переваги: простий текстовий інтерфейс, зрозумілий для новачків; добре читається навіть при прямому сонячному освітленні.

Недоліки: не підтримує графіку, лише 2 рядки по 16 символів; без I^2C модуля займає багато пінів Arduino (до 6–8), що ускладнює підключення інших модулів.

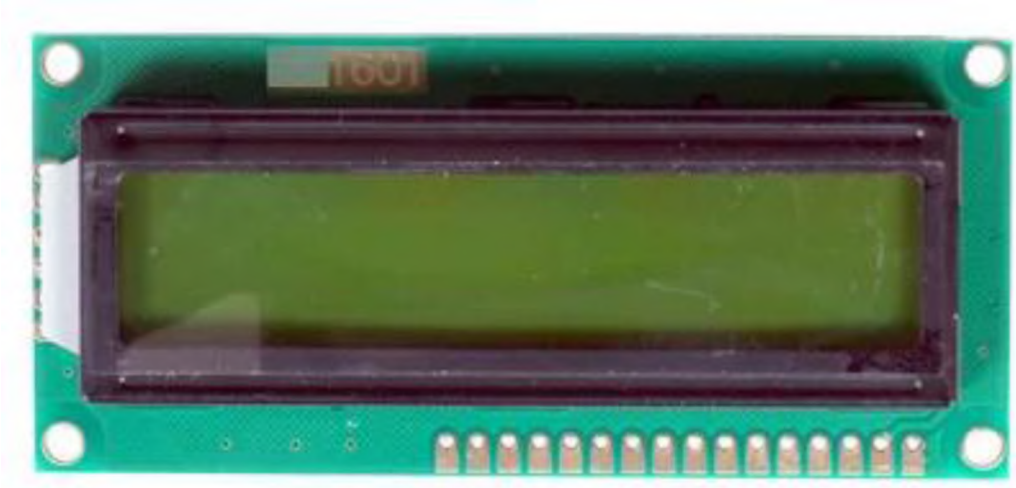


Рисунок 1.10. Приклад LCD 16x2

Для зручного сприйняття інформації створено табл.1.2., де порівняно моделі дисплеїв:

Таблиця 1.2. Порівняльний аналіз дисплеїв

Параметр	OLED 0.96" (SSD1306)	TFT 1.8" (ST7735)	LCD 16x2 (HD44780)
Тип дисплея	OLED (монохромний)	TFT (кольоровий)	LCD (текстовий)
Роздільна здатність	128×64	160×128	16 символів × 2 рядки
Інтерфейс підключення	I2C	SPI	Паралельний / I2C
Споживання енергії	~20 мА	~50–60 мА	~1–5 мА
Підтримка графіки	Так	Так	Ні
Розміри	Компактний. Підходить для невеликих проєктів	Більший екран, більше можливостей візуалізації	Середній, але не дозволяє графіки
Контрастність	Висока Пікселі світяться самостійно	Середня Потребує підсвічування	Середня Потребує підсвічування
Складність підключення	Низька Два дроти I2C.	Середня Потрібно більше проводів	Висока без модуля I2C

Після проведення аналізу було обрано OLED 0.96" (SSD1306) тому що: він забезпечує зручне підключення, якісне відображення, та дозволяє виводити дані у зручному для читання вигляді.

1.5 Дослідження програмних засобів для збору апаратних даних ПК

Для реалізації функції діагностики стану ПК у реальному часі важливим кроком є вибір програмного забезпечення, яке надає доступ до системних апаратних показників. У цьому розділі буде виконано ґрунтовне дослідження та порівняльний аналіз кількох програмних засобів моніторингу ПК, а саме: Диспетчер завдань Windows (Task Manager), Open Hardware Monitor, MSI Afterburner. А також обґрунтування вибору найбільш відповідного програмного засобу моніторингу ПК для забезпечення стабільної роботи пристрою при мінімальних витратах та максимальній простоті реалізації.

Програмні засоби моніторингу ПК:

Диспетчер завдань Windows (Task Manager).

Це базовий інструмент, вбудований у Windows, який надає загальну інформацію про використання CPU, RAM, диска та мережі.

Переваги: Вбудований, завжди доступний.

Недоліки: Обмежена деталізація (особливо щодо GPU), вимагає постійного відкритого вікна на екрані, не дає «з першого погляду» візуальної оцінки.

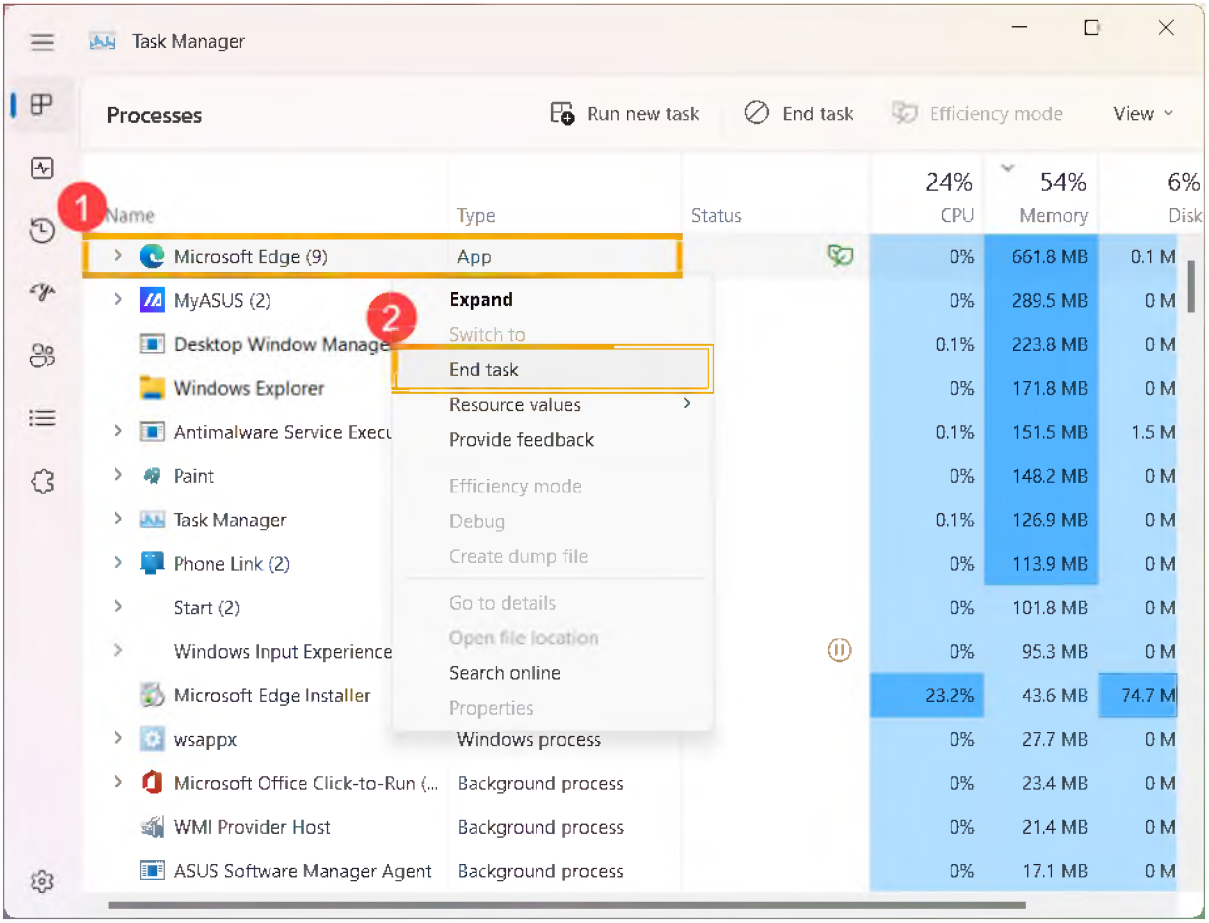


Рисунок 1.11. Приклад роботи Диспетчер завдань Windows

Open Hardware Monitor.

Безкоштовна, дуже популярна утиліта, що надає глибоку інформацію про всі апаратні компоненти: температури, напруги, навантаження CPU/GPU, швидкість вентиляторів.

Переваги: Вичерпна інформація, точність даних.

Недоліки: Вимагає активного вікна на екрані, може бути складною для швидкого перегляду, не завжди зручна для постійного моніторингу під час гри чи роботи.

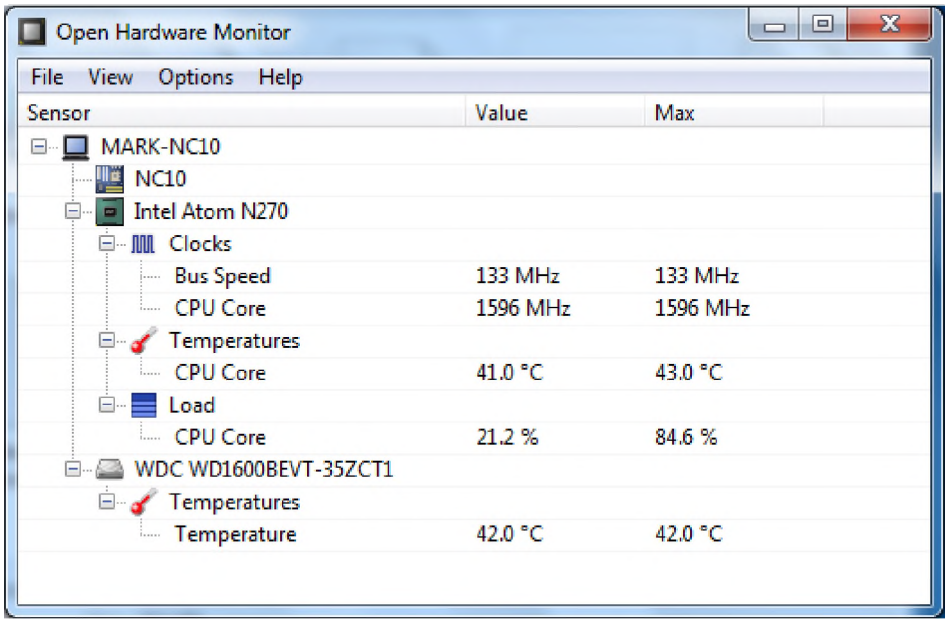


Рисунок 1.12. Приклад роботи Open Hardware Monitor

MSI Afterburner.

Ця програма, окрім моніторингу, пропонує функції розгону відеокарти та управління вентиляторами. Вона має графічний інтерфейс та можливість відображення оверлею (накладання інформації поверх ігор).

Переваги: Детальний моніторинг, функції керування, візуальна привабливість, оверлей.

Недоліки: Орієнтовані на геймерів, вимагають встановлення значного програмного забезпечення, можуть бути ресурсоємними.

Рисунок 1.13. Приклад роботи MSI Afterburner

Для зручного сприйняття інформації створено табл.1.3., де порівняно програмних засобів моніторингу ПК

Таблиця 1.3. Порівняльний аналіз програмних засобів моніторингу ПК

Назва	Тип	Переваги	Недоліки	Висновок
Task Manager	Програмна утиліта	Вбудований, простий у доступі	Обмежена деталізація, не зручно моніторити під час ігор	Підходить для базового перегляду стану
MSI Afterburner	Інструмент моніторингу/ розгону	Оверлей, детальний моніторинг, управління	Ресурсоємний, орієнтований на геймерів	Підходить для геймерів
Open Hardware Monitor	Open Source утиліта	Безкоштовний, відкритий код, JSON API	Старий інтерфейс, потребує налаштування	Ідеально для інтеграції в проєкти

Після проведення аналізу було вибрано Open Hardware Monitor, тому що:

Надійність і точність даних

Має високу точність зчитування інформації з апаратних сенсорів, підтримує широкий перелік материнських плат, процесорів, відеокарт, і точно передає критично важливі для обраних для проєкта показників: завантаження процесора (CPU), графічного процесора (GPU) та оперативної пам'яті (RAM).

Простота інтеграції

Не потребує складного інтерфейсу, постійної присутності на екрані чи важких GUI-складових, а це дозволяє створити компактний скрипт на Python, що зчитує дані з його інтерфейсу та передає їх на Arduino.

Мала ресурсоємність

Споживає мінімальну кількість системних ресурсів, не впливає на продуктивність, та може працювати у фоновому режимі, що особливо важливо для старіших ПК чи під час навантаження системи.

Відсутність реклами і надлишкового функціоналу

Не включає маркетингових елементів, розгону та RGB-контролю — що робить його більш стабільним та надійним для чисто діагностичних цілей.

Простий інтерфейс для тестування та відладки.

У процесі налагодження проєкту важливо мати змогу візуально перевіряти відображення даних, а даний ПЗ дозволяє це зробити зручно й швидко без додаткових модулів та надбудов.

З огляду на це можна зробити висновки, що Open Hardware Monitor є практичним і ефективним вибором для реалізації проєкту моніторингу ПК з використанням Arduino.

1.6 Розробка моделі пристрою і визначення базових параметрів

Проєктований пристрій призначений для зручної візуалізації поточного стану компонентів ПК у режимі реального часу. Основне його завдання– надання чітких даних про навантаження центрального процесору (CPU), графічного процесору (GPU) та оперативної пам'яті (RAM). Завдяки чому користувач може оперативно оцінювати стан ПК без необхідності кожен раз відкривати додаткове програмне забезпечення на моніторі ПК.

1.6.1 Розробка структурної схеми пристрою діагностики

Було розроблено проєкт, який передбачає тісну інтеграцію між апаратною (Arduino Nano, дисплей, зв'язок з ПК) та програмною (Python скрипт, Open Hardware Monitor) частинами. Для забезпечення логічної та фізичної взаємодії я створив структурну схему. Структурна схема дозволяє чітко візуалізувати всі зв'язки між компонентами та зрозуміти їх роль у роботі пристрою.

Базова архітектура складається з трьох основних блоків:

Сенсорна частина (програмний моніторинг): програмне забезпечення Open Hardware Monitor, яке працює у фоновому режимі на ПК, забезпечує постійний збір показників про завантаження процесора (CPU), графічного процесора (GPU) та оперативної пам'яті (RAM). Всі ці параметри доступні через веб-інтерфейс OHM у форматі JSON.

Програмний застосунок OHM, працює у фоновому режимі на ПК, та робить постійний збір даних про навантаження системи за різними показниками. Всі ці

дані доступні через локальний веб-сервер (http://localhost:8085/data.json) у форматі (JSON). Це дозволяє не створювати фізичні файли які займають місце у сховищі ПК.

Передавальний блок (Python-скрипт): окрема програма, написана мовою Python, яка виконує періодичне опитування локального веб-сервера OHM (http://localhost:8085/data.json). Отримані дані обробляються, фільтруються, перетворюються у формат, придатний для Arduino (наприклад: CPU:32 GPU:28 RAM:65) і передаються через COM-порт за допомогою модуля PySerial.

Також обрані дані виводяться у командну стрічку що дозволяє, за бажанням, переглянути дані, які ПК відправляє на Arduino. Після цього отримані дані передаються на саму Arduino через COM-порт за допомогою модуля PySerial.

Апаратна частина (Arduino + OLED-дисплей): Arduino Nano отримує текстову інформацію через послідовний порт, парсить її та виводить у зручному вигляді на OLED-дисплей розміром 128×64 пікселів. Дисплей працює за інтерфейсом I^2C, що мінімізує кількість необхідних з'єднань (використовуються лише SDA та SCL лінії).

Arduino Nano отримує інформацію через послідовний порт, парсить її (розбирає рядок на окремі компоненти, наприклад, «CPU», «XX», «GPU», «YY», «RAM», «ZZ», та перетворює їх на числові значення) та виводить на OLED-дисплей у зручному форматі. Дисплей працює за інтерфейсом I^2C, таке підключення мінімізує кількість необхідних з'єднань.

Взаємодія Користувача: користувач взаємодіє з пристроєм не на пряму, а шляхом перегляду відображених даних на OLED-дисплеї. Аналізуючи навантаження своєї системи користувач може виявляти аномалії та, якщо потрібно, вживати відповідних заходів (наприклад, закрити ресурсоємні програми, перевірити на наявність вірусів і т.п.).

На структурній схемі ці три компоненти зображені як окремі блоки, з'єднані стрілками, що вказують напрямок потоку даних:

- ПК → Open Hardware Monitor (відправляє дані у локальний веб-сервер);
- Open Hardware Monitor → Python-скрипт (веб-запит до OHM);

– Python-скрипт → Arduino Nano (через COM-порт);
– Arduino Nano → OLED-дисплей (через I^2C).

Рисунок 1.14. Структурна схема пристрою діагностики

Зображена схема сформована для швидкого орієнтування у функціональному поділі системи.

1.6.2 Розробка алгоритму збирання та передачі діагностичних даних

У цьому підрозділі буде описано покроковий алгоритм дії функціонування пристрою.

Робота пристрою: далі представлено покроковий опис алгоритму, який реалізується на рівні програмного забезпечення та мікроконтролера: функціонування пристрою ґрунтується на циклічному зборі, обробці, передачі та візуалізації даних.

Спочатку запускається Open Hardware Monitor: програма, яка запускається у фоновому режимі та відкриває доступ до інформації про стан системи через вбудований інтерфейс, та надає доступ до локального HTTP-серверу (localhost:8085/data.json). Програма автоматично оновлює значення сенсорів у JSON-структурі, тому користувачеві не потрібно взаємодіяти з інтерфейсом OHM після його запуску.

Потім відбувається збір даних через Python-скрипт, він реалізується за допомогою бібліотек requests (для запитів HTTP) та PySerial (для серійної передачі). Кожну секунду програма звертається до адреси OHM та отримує JSON-дані потім шукає у структурі значення CPU Total, GPU Core (у розділі Load), та Memory Load. Ці значення очищуються від символів %, замінюються коми на крапки та перетворюються у формат чисел з плаваючою точкою.

Дані зберігаються у змінних і конвертуються у форматований рядок (f"CPU:{cpu} GPU:{gpu} RAM:{ram}") і відправляються в COM-порт і Arduino отримує лише відфільтровані значення необхідні для правильного відображення

даних.

Далі здійснюється прийом та обробка Arduino:

На стороні Arduino діє нескінченний цикл, у якому зчитується інформація з COM-порту по одному символу, коли зчитано символ нового рядка \n, програма розпізнає це як завершення пакету. Далі починається операція обробки: Arduino парсить рядок, виділяє значення CPU, GPU, RAM та виводить їх на дисплей у обраному мною вигляді.

Після починається виведення даних на OLED-дисплей, який отримує команди від Arduino через інтерфейс I^2C. Всі три параметри виводяться у текстовому форматі великим шрифтом для полегшеного читання з відстані:

Load PC:

CPU: 35%

GPU: 12%

RAM: 48%

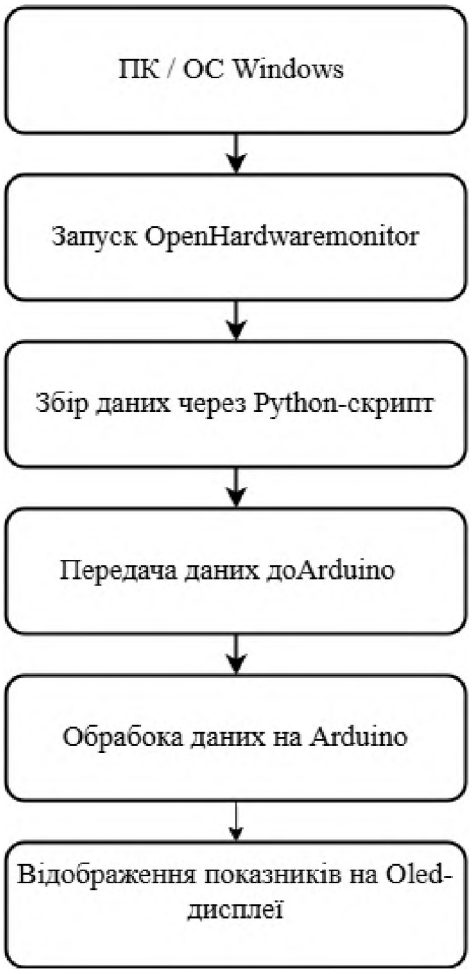


Рисунок 1.15. Структурна схема пристрою діагностики

Зображена схема сформована для наглядної ілюстрації послідовності дій.

Дане рішення характеризується простою реалізацією, гнучкістю в налаштуваннях і можливістю масштабування, що робить алгоритм надійним, доступним і гнучким, у ньому, при необхідності, можливо змінити представлені параметри чи змінити періодичність оновлення даних.

1.6.3 Визначення базових технічних параметрів

А у цьому підрозділі буде описано базові технічні характеристики, які доцільні для стабільного та ефективного функціонування пристрою. Ці параметри стосуються як апаратної частини, так і програмної логіки.

Частота оновлення:

Щосекундна частота оновлення — є достатньою для візуального контролю, але не створює надмірного навантаження на мікроконтролер чи канал зв'язку.

Точність показників:

Для спрощення формату обміну даними та зниження ризиків помилок в Arduino будуть використовуватися цілі значення (%), округлені з плаваючої точки.

Енергоспоживання:

Ця система легко живиться від USB-порту ПК, тому що Arduino Nano споживає близько 30-40 мА у звичайному режимі, а OLED-дисплей додає ще 10-15 мА.

Характеристики та функціональність OLED-дисплея SSD1306:

Роздільна здатність 128×64 пікселів дозволяє вивести 4 рядки тексту по 16 символів великим шрифтом. Контролер SSD1306 підтримується популярними бібліотеками Adafruit.

Формат переданих даних:

CPU:22 GPU:12 RAM:38\n

Рядок має фіксований формат, де параметри розділені пробілами та закінчуються символом нового рядка.

Параметри передачі даних:

Послідовне з'єднання має швидкість 115200 бод (приблизно 11 520 байт/сек). Це дозволяє швидко передавати дані без затримок, втрат та перевантаження буфера Arduino.

Перспективи розширення набору параметрів:

До системи легко додати нові параметри (наприклад, температура SSD, навантаження VRAM та інш.), розширивши скрипт і адаптувавши Arduino.

Зазначені вище параметри гарантують стабільність та ефективність роботи системи моніторингу.

1.6.4 Склад і взаємозв'язок програмних та апаратних компонентів

Для повного функціонування системи діагностики ПК було обрано оптимальний набір апаратних і програмних компонентів. Їх взаємозв'язок можна умовно поділити на три рівні: програмний, інтерфейсний і апаратний.

Апаратні компоненти:

Arduino Nano — основний обчислювальний елемент, що приймає дані через UART (апаратний модуль Arduino, що забезпечує послідовну передачу даних і використовується для зв'язку з комп'ютером через COM-порт) та виводить їх через I^2C (двопровідний інтерфейс: SDA, SCL).

OLED-дисплей 128×64 SSD1306 — засіб відображення інформації для користувача.

USB-кабель — передає живлення та дані між ПК та Arduino.

Програмні компоненти:

Open Hardware Monitor (OHM) — безкоштовна утиліта для збору показників стану ПК. Працює як локальний веб-сервер з JSON-виводом.

Python-скрипт — посередник між OHM і Arduino. Забезпечує обробку даних та передачу в потрібному форматі через COM-порт.

Arduino Sketch — прошивка, яка обробляє вхідні рядки, парсить значення та виводить їх на дисплей.

Інтерфейсний зв'язок:

COM-порт (Serial): зв'язок між ПК та Arduino для передачі рядків із даними.

I^2C: з'єднання між Arduino та OLED-дисплеєм. Мінімум проводів, стабільна швидкість.

Принцип взаємодії:

OHM → надає дані JSON.

Python → обробляє JSON, надсилає текстовий рядок через COM.

Arduino → зчитує рядок, обробляє, виводить на дисплей.

Усі ці компоненти необхідні для забезпечення коректної роботи і діють як єдина інтегрована система, де кожен елемент мають чітко визначені функції. Такий модульний підхід спрощує процес розширення функціональності та гарантує зручність налагодження та дозволяє оновлювати окремі компоненти без необхідності значної перебудови системи.

Систему можна легко доповнювати новими модулями, встановлювати додаткові сенсори, змінювати методи обробки даних, все це досягається завдяки гнучкій архітектурі. Це дає змогу покращуючи її продуктивність та функціональність, а також можливість адаптувати систему до майбутніх задач, що забезпечує стабільну роботу в різних умовах

1.7 Підбір компонентів пристрою та обґрунтування вибору

Щоб створити ефективний прилад для діагностики стану ПК потрібно розуміти принцип роботи пристрою та ретельно підібрати основні апаратні та програмні компоненти так, щоб вони ефективно взаємодіяли між собою. Вибір елементів було обумовлено доступністю, ціною, технічними вимогами та сумісністю з іншими складовими. Центральними елементами було обрано мікроконтролер Arduino Nano та OLED-дисплей SSD1306 0.96”, як найбільш оптимальні. Також буде створено програмне забезпечення, що забезпечує взаємодію компонентів.

1.7.1 Апаратна частина

1. Мікроконтролер Arduino Nano

Arduino Nano — компактний мікроконтролер на базі ATmega328P, обраний через свої невеликі розміри, низьке енергоспоживання та сумісність із численними дисплейними модулями.

Плата Arduino побудована на базі мікроконтролера Atmel AVR, а також елементів обв’язки для програмування та інтеграції з іншими пристроями. На багатьох платах наявний лінійний стабілізатор напруги +5V та +3,3V. Тактування

здійснюється на частоті 16 та 8 МГц кварцовим резонатором. У мікроконтролер записаний завантажувач bootloader (невелика програма, що дозволяє завантажувати користувацький код через USB-з'єднання), тому зовнішній програматор не потрібен.

Основні параметри мікроконтролера Arduino Nano представлено в табл.1.4.:

Таблиця 1.4. Основні параметри мікроконтролера Arduino Nano

Параметр	Значення
Процесор	ATmega328P
Тактова частота	16 МГц
Кількість цифрових пінів	14 (з них 6 — PWM)
UART	Є (Serial TX/RX)
Живлення	5 V (USB)
Пам'ять	32 КБ Flash, 2 КБ SRAM
Підключення по USB	через mini-USB кабель

Arduino приймає текстові команди (напр. CPU:45 GPU:37 RAM:60) і виводить значення на OLED-дисплей чи інший інтерфейс.

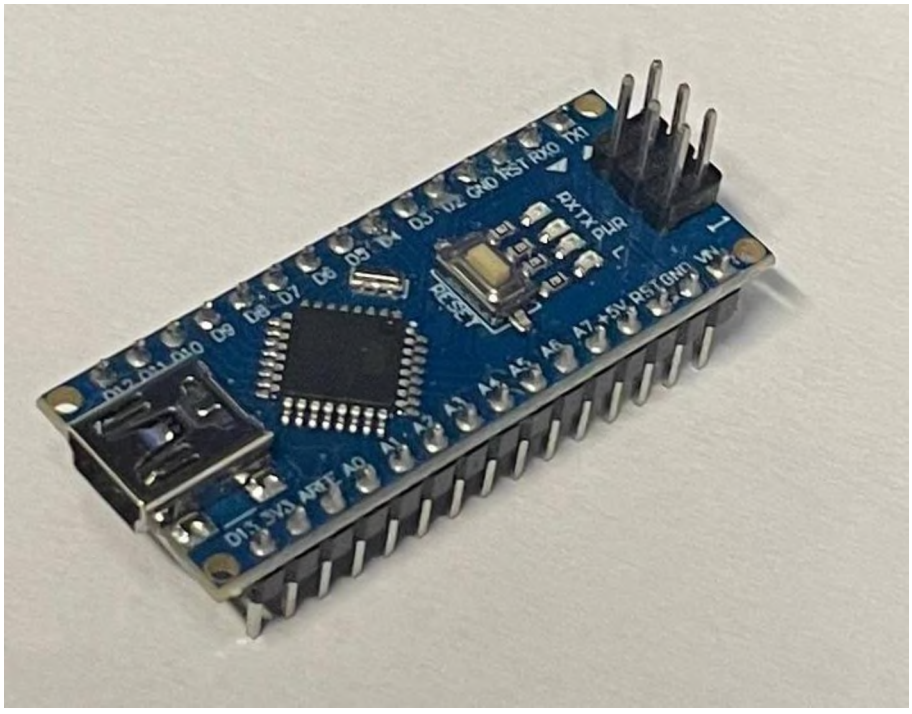


Рисунок 1.16. Фото мікроконтролера, що використано в проєкті

2. OLED-дисплей 0.96" (SSD1306)

OLED (Organic Light Emitting Diode) дисплеї є відмінним вибором для компактних проєктів завдяки своїй високій контрастності (кожен піксель світиться сам по собі), низькому енергоспоживанню та широким кутам огляду. SSD1306 – це поширений контролер для таких дисплеїв.

Принцип роботи: OLED-дисплей SSD1306 керується за допомогою протоколу I²C (Inter-Integrated Circuit). Цей протокол є послідовним двопровідним інтерфейсом, що складається з лінії даних (SDA) та лінії тактування (SCL). Мікроконтролер (Arduino Nano) виступає в ролі «майстра» і відправляє команди та дані на дисплей, який є «підлеглим» пристроєм. Дисплей має власну вбудовану пам'ять, яка зберігає зображення, що має бути відображене. Коли Arduino надсилає дані, контролер SSD1306 оновлює цю пам'ять, і пікселі дисплея відповідно змінюють свій стан (вмикаються/вимикаються). Для реалізації графічного виведення інформації на OLED-дисплейв Arduino IDE використовується набір спеціалізованих бібліотек, таких як:

Wire — це стандартна бібліотека для роботи з інтерфейсом I²C в Arduino, яка дозволяє реалізовувати обмін даними між мікроконтролером і зовнішніми пристроями (сенсорами, дисплеями, модулями тощо) за двопровідним протоколом I²C. Бібліотека може бути як в режимі ведучого (master), так і в режимі підлеглого (slave). У контексті даного проєкту бібліотека Wire використовується для зв’язку між Arduino Nano та OLED-дисплеєм SSD1306 по інтерфейсу I²C.

Adafruit_SSD1306 — це бібліотека для монохромних OLED-дисплеїв на основі SSD1306. Вона підтримує I²C та SPI інтерфейси, що дозволяє легко інтегрувати дисплей у проєкти з Arduino, ESP32, ESP8266 та іншими мікроконтролерами.

Adafruit_GFX — це основна графічна бібліотека для дисплеїв Adafruit, яка забезпечує малювання тексту, ліній, фігур та зображень. Вона використовується разом із апаратно-специфічними бібліотеками, такими як Adafruit_SSD1306 чи Adafruit_ST7735, для роботи з конкретними дисплеями.

Ці бібліотеки абстрагують низькорівневі операції I²C та надають зручні

функції для малювання тексту, ліній, прямокутників та інших графічних елементів. Також, усі ці бібліотеки добре інтегруються між собою та регулярно оновлюються

Основні параметри OLED-дисплея SSD1306 представлено в табл.1.5.:

Таблиця 1.5. Основні параметри OLED-дисплея SSD1306

Параметр	Значення
Тип	OLED 0.96"
Контролер	SSD1306
Роздільна здатність	128×64 пікселів
Інтерфейс	I^2C (4 контакти: VCC, GND, SDA, SCL)
Напруга живлення	3.3–5 V
Споживана потужність	~20 мА

Дисплей показує три основні значення: CPU, GPU та RAM у відсотках у реальному часі.

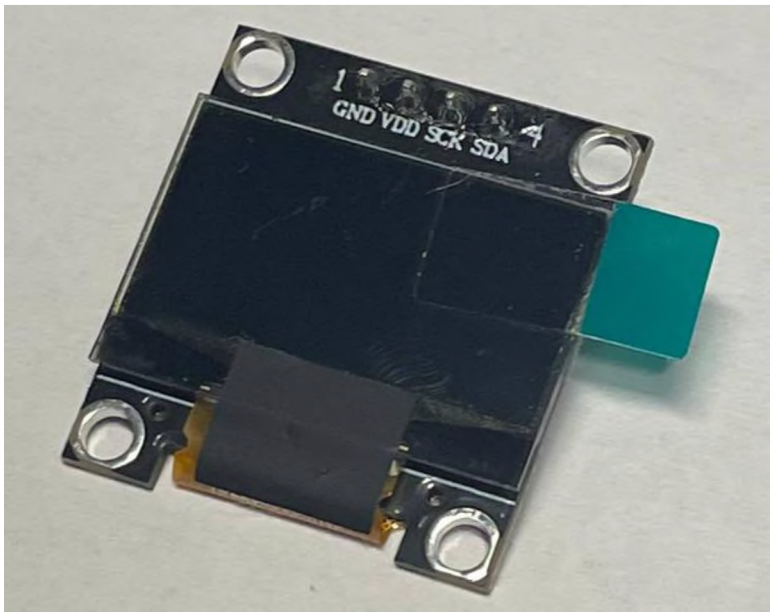


Рисунок 1.17. Фото дисплея, що використано в проєкті

3. Кабелі Dupont (Мати-Мати)

Кабелі Dupont – це стандартні з'єднувальні кабелі, що використовуються для прототипування та з'єднання електронних компонентів.

Тип: «Мати-Мати». Це означає, що обидва кінці кабелю мають роз'єми типу «мама», призначені для підключення до пінів (штирів) компонентів.

Використовуються для створення, як тимчасових, так і постійних з'єднань між Arduino Nano та OLED-дисплеєм. Сумісні зі стандартними гребінками пінів на більшості електронних модулів. Дозволяють швидко і легко з'єднати компоненти без використання паяльника, що ідеально підходить для розробки та тестування пристрою.

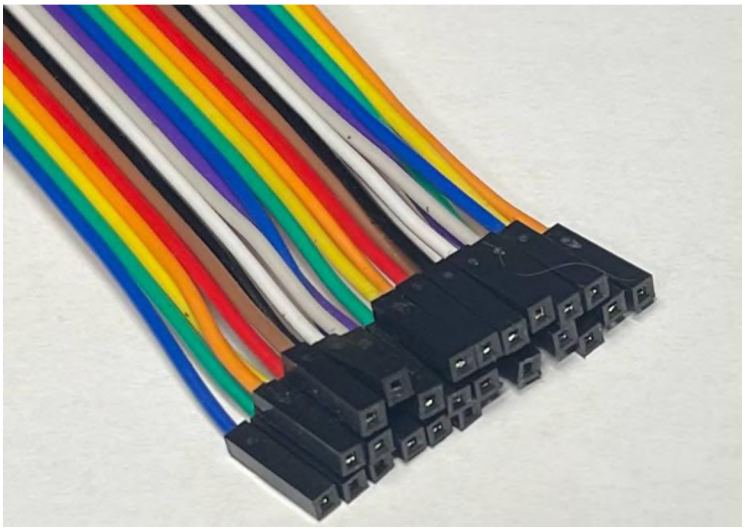


Рисунок 1.18. Фото кабелів, що використано в проєкті

1.7.2 Програмні компоненти

1. Open Hardware Monitor

Це безкоштовне програмне забезпечення з відкритим кодом, яке дозволяє отримувати діагностичні дані з різних сенсорів ПК: температури, навантаження процесора та графіки, рівня використання пам’яті тощо.

Основні параметри, описано в табл.1.6.:

Таблиця 1.6. Основні параметри використання Open Hardware Monitor у проєкті

Параметр	Значення
Версія	0.9.6 (використовується у проєкті)
Сумісність	Windows XP/7/10/11
Формат експорту даних	JSON через локальний HTTP-сервер
Частота оновлення	до 1 разу на секунду
Підтримка сенсорів	CPU, GPU, RAM, HDD/SSD, материнська плата

OHM виступає джерелом даних для Python-скрипта, який здійснює обробку та передає ці значення на Arduino.

2. Python-скрипт (модуль збору і передачі даних)

Python використовується як джерело зв'язку між Open Hardware Monitor та мікроконтролером Arduino.

Мова програмування Python була обрана завдяки своїй кросплатформенності, великій кількості бібліотек та простоті синтаксису. Також Python дозволяє легко інтегруватися з операційною системою та отримувати доступ до широкого спектра системних метрик, які було б вкрай складно чи неможливо отримати безпосередньо з Arduino (наприклад, навантаження CPU/GPU з точки зору ОС).

Основні бібліотеки, які були використані у Python-скрипті:

– бібліотека Requests — є одним із найпопулярніших інструментів у Python для роботи з HTTP-запитами. Вона дозволяє відправляти GET, POST, PUT, DELETE та інші види запитів до веб-серверів, отримувати відповіді та обробляти їх у зручному форматі.

Однією з ключових переваг Requests є простота використання – бібліотека автоматично обробляє кодування, заголовки та можливі помилки. Вона широко використовується для API-запитів, веб-скрапінгу та взаємодії зі сторонніми онлайн-сервісами.

Також Requests підтримує автентифікацію, роботу з куками та сесіями, що важливо для взаємодії зі складними веб-додатками. Завдяки зручному синтаксису, розробники можуть легко працювати з мережевими запитами без необхідності налаштовувати низькорівневі параметри.

– бібліотека PySerial необхідна для взаємодії Python-програм із серійними портами. Вона дозволяє читати та записувати дані через COM-порт, що є критично важливим при роботі з мікроконтролерами, такими як Arduino.

Завдяки PySerial можна налаштовувати параметри з'єднання, включаючи швидкість передачі (baud rate), кількість бітів даних, парність та стоп-біти. Вона підтримує автоматичне відкриття/закриття портів, тайм-аути та буферизацію переданих даних.

Однією з головних переваг PySerial є простота інтеграції – достатньо лише кількох рядків коду для встановлення зв'язку та обміну даними між комп'ютером та мікроконтролером. Це робить PySerial ідеальним вибором для проєктів, що потребують управління апаратними пристроями через USB чи COM-порт.

Отримання JSON-файлу з http://localhost:8085/data.json.

Парсинг значень CPU, GPU, RAM.

Передача даних через COM-порт (Serial) на Arduino зі швидкістю 115200 бод.

Основні параметри, описано в табл.1.7.:

Таблиця 1.7. Основні параметри використання Python у проєкті

Параметр	Значення
Версія Python	3.x (використовується у проєкті)
Сумісність	Windows XP/7/10/11, Linux, MacOS
Формат обміну даними	JSON через HTTP-запит (requests), передача через COM-порт (pyserial)
Частота обміну	до 1 раз на секунду
Ключові бібліотеки	Requests, PySerial
Роль у проєкті	Посередник між Open Hardware Monitor та Arduino

1.8 Розробка схеми підключення елементів пристрою

Процес розробки проєктованого пристрою передбачає не тільки вибір оптимальних компонентів, а й детальне планування схеми їх підключення та взаємодії. Оскільки це ключовий етап, що визначає стабільність, ефективність та надійність роботи системи.

У цьому розділі буде описано детальний метод підключення основних апаратних модулів: мікроконтролера Arduino Nano до OLED-дисплея SSD1306 за допомогою кабелів Dupont, для живлення від комп'ютера та зв'язку з ним через COM-порт.

У процесі проведення попередніх досліджень було розроблено функціональну схему пристрою для моніторингу системних параметрів ПК:

1. Підключення ПК до Arduino

Для взаємодії між комп'ютером і Arduino використовується кабель USB → mini USB, який виконує подвійну функцію:

Живлення плати Arduino (3V)

Передача даних через UART (Serial)

Мікроконтролер Arduino Nano мають вбудований апаратний UART, що дозволяє здійснювати прямий обмін даними через порти RX/TX. Завдяки USB-UART перетворювачу, пристрій зв'язується з ПК та отримує інформацію без додаткових зовнішніх модулів.

У програмному коді Arduino використовується команда:

```
Serial.begin(115200);
```

Рисунок 1.19. Фрагмент коду для встановлення серійної комунікації

Це забезпечує ініціалізацію послідовного порту зі швидкістю 115200 бод, що гарантує швидку та стабільну передачу даних без затримок.

2. Підключення OLED-дисплея SSD1306

Для графічного інтерфейсу використовується OLED SSD1306, який взаємодіє з Arduino через протокол I²C. Основною перевагою цього протоколу є мінімальна кількість проводів (лише два), необхідних для обміну даними:

– A4 (SDA) +– канал передачі інформації;

– A5 (SCL) – канал синхронізації.

OLED-дисплей підключається до відповідних виводів Arduino таким чином:

– SDA → A4 Передача даних (I²C);

– SCK → A5 Синхронізація (I²C);

– VDD → 3V3 Живлення дисплея;

– GND → GND Земля (спільний контур).

На рисунку 1.20. показано детальну схему фізичного підключення OLED-дисплея на базі контролера SSD1306 до мікроконтролерної плати Arduino Nano, яка використовується в проєкті для виведення діагностичної інформації, отриманої з Open Hardware Monitor.

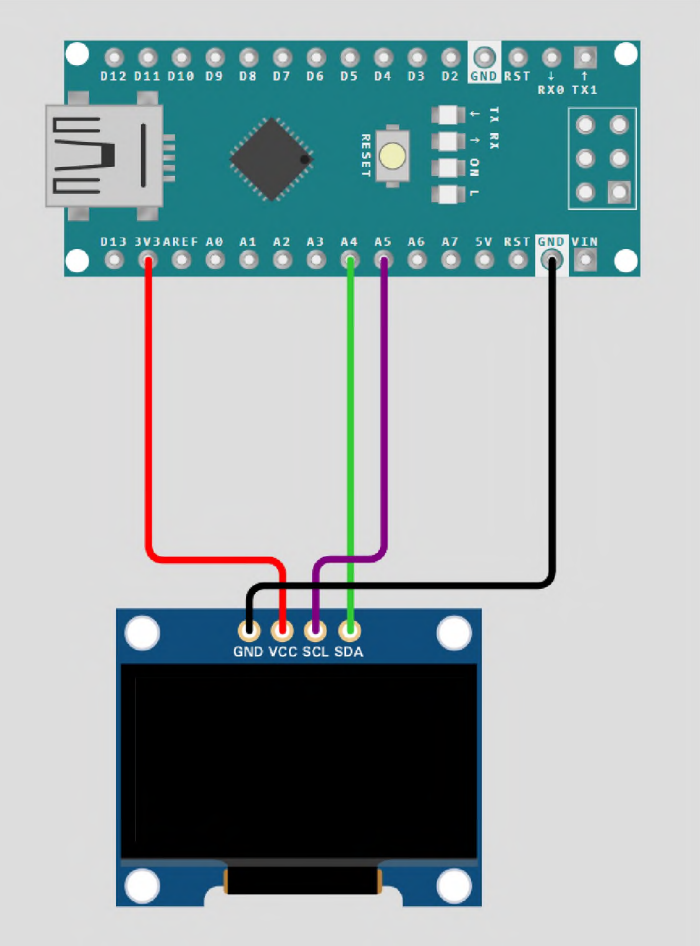


Рисунок 1.20. Схема підключення OLED-дисплея SSD1306 до Arduino Nano

Схема підключення забезпечує стабільне з'єднання та передачу даних між мікроконтролером Arduino Nano та OLED-дисплеєм через інтерфейс I²C, що дозволяє в реальному часі відображати параметри системного моніторингу, отримані з Open Hardware Monitor.

На I²C-шині OLED-дисплей має фіксовану адресу – 0x3C, яка вказується у коді:

```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
```

Рисунок 1.21. Адресація OLED-дисплея на I²C-шині в коді Arduino

Використання цієї команди забезпечує коректну ініціалізацію дисплея, що дозволяє виводити CPU, GPU та RAM у реальному часі.

3. Електроживлення

Живлення Arduino здійснюється безпосередньо через USB-кабель (3V), що

одночасно підтримує обмін даними. Дисплей SSD1306 також отримує живлення від пінів 3V3 та GND самої плати Arduino, що спрощує підключення та унеможливлює конфлікти напруги.

Загальне енергоспоживання пристрою не перевищує 50 мА, тому для роботи не потрібне зовнішнє живлення. Достатньо підключення до ПК.

Описана схема підключення компонентів забезпечує стабільний обмін даними, енергоефективність та простоту інтеграції. Використання I^2C-протоколу для OLED-дисплея мінімізує потребу у додаткових проводах, а живлення від USB гарантує безперебійну роботу пристрою. Обрані компоненти мають високу сумісність між собою, а це спрощує налаштування та зменшує ризик апаратних помилок. Завдяки оптимальному використанню ресурсів Arduino і ефективної передачі даних від ПК буде реалізовано компактну систему моніторингу в реальному часі.

1.9 Реалізація програмного забезпечення

Програмне забезпечення для пристрою моніторингу стану ПК, розробленого на базі мікроконтролера Arduino Nano, складається з двох частин: комп'ютерної (Python-скрипт) та мікроконтролерної (Arduino-скетч на C++).

Основна мета – надати користувачеві візуальну індикацію продуктивності ПК у реальному часі без необхідності постійно відкривати програму моніторингу на екрані монітора. Така реалізація поєднує простоту апаратної частини з гнучкістю програмної логіки, дозволяючи швидко адаптувати пристрій до різних сценаріїв використання.

Головним завданням цього програмного забезпечення є прийом даних про навантаження процесора, відеокарти та оперативної пам'яті з персонального комп'ютера, їх обробка, а також відображення на зовнішньому OLED-дисплеї. Комунікація між комп'ютером і Arduino здійснюється через COM-порт із використанням протоколу USB.

Основні етапи розробки програмного забезпечення:

Комп'ютерна частина ґрунтується на використанні Open Hardware Monitor (OHM) — безкоштовного програмного забезпечення з відкритим кодом, яке

дозволяє отримувати діагностичні дані з сенсорів ПК: температуру процесора та графічного адаптера, завантаження системи, обсяг використаної оперативної пам'яті, стан накопичувачів, тощо. У проєкті використовується версія OHM 0.9.6, яка підтримує експорт даних у форматі JSON через локальний HTTP-сервер, із частотою оновлення до одного разу на секунду. Python-скрипт звертається до цього локального сервера, обробляє отримані JSON-дані, виділяє потрібні параметри й надсилає їх через COM-порт у вигляді форматованого тексту, придатного для обробки мікроконтролером.

Мікроконтролерна частина, реалізована у вигляді Arduino-скетча, приймає дані через послідовний інтерфейс USB (Serial), розбирає отримані повідомлення та виводить їх на OLED-дисплей з роздільною здатністю 128×64 пікселі, побудований на базі контролера SSD1306. Для зв'язку з дисплеєм використано інтерфейс I^2C, з підключенням виводів SDA та SCK до аналогових пінів A4 та A5 відповідно. Ця схема забезпечує надійний обмін даними та відображення параметрів продуктивності ПК.

Живлення OLED-дисплея подається від Arduino Nano (через 3.3V). Усі фізичні з'єднання реалізовані відповідно до стандартної I^2C схеми, це спрощує підключення та забезпечує сумісність із великою кількістю компонентів. Схема є мінімалістичною, але й ефективною, і дозволяє забезпечити стабільну роботу системи без потреби у складній периферії.

Зображена схема алгоритму на рисунку 1.22. сформована для відображення послідовності виконання дій.

Ключові особливості алгоритму:

Надійність: передбачено обробку помилок та автоматичне очікування при втраті зв'язку і повторна спроба зчитування.

Автономність: після одноразового запуску системи користувач не має необхідності здійснювати жодних дій. Пристрій працює у постійному циклі опитування та оновлення даних.

Мінімальне споживання ресурсів: алгоритм адаптований до обмежених можливостей мікроконтролера.

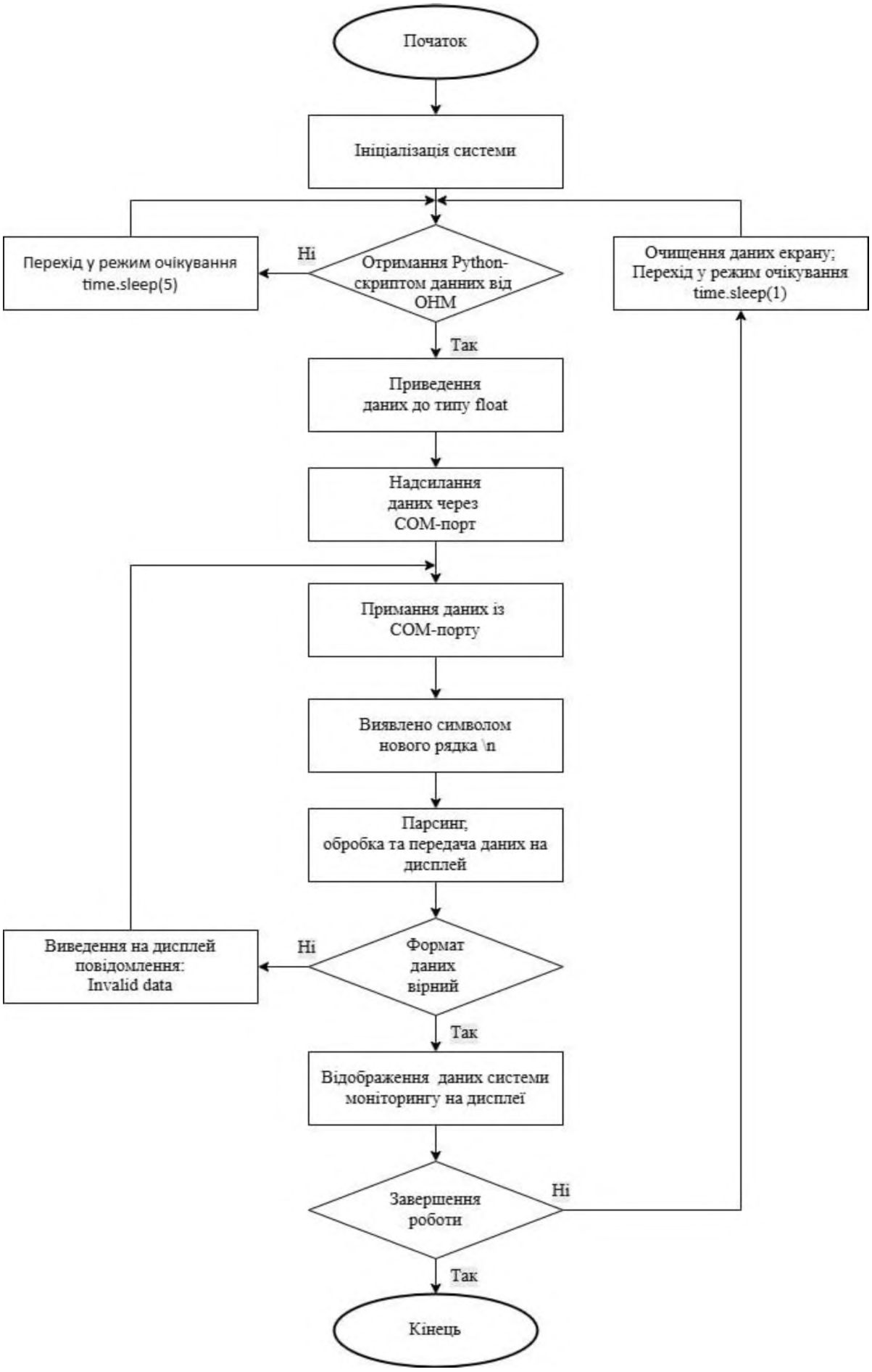


Рисунок 1.22. Узагальнена блок-схема алгоритму вимірювань

1.9.1 Розробка Python-скрипта: збір та передача даних з OHM

Під час розробки проєкту однією з ключових частин є програмна реалізація збору актуальної діагностичної інформації про навантаження на основні компоненти ПК, її обробка та передача у пристрій на базі Arduino Nano для подальшого візуального відображення на зовнішньому OLED-дисплеї. Для цієї задачі був використаний Python-скрипт, він відповідає за підключення до локального веб-сервера, створеного програмою Open Hardware Monitor і передачу відфільтрованих даних у Arduino через COM-порт.

Python було обрано як основну мову програмування у цьому завданні через його простоту синтаксису, широку підтримку бібліотек для роботи з мережею та послідовними інтерфейсами, а також можливістю легкого розгортання на будь-якому комп'ютері.

Встановлення бібліотек:

Для повноцінної роботи скрипта потрібно встановити дві основні бібліотеки:

- Requests — для HTTP-запитів до сервера OHM.
- PySerial — для зв'язку з COM-портом.

Команда для встановлення бібліотек:

```
>pip install requests pyserial
```

Отримання даних з OHM:

Open Hardware Monitor передає дані (формат JSON), які доступні через локальну адресу:

http://localhost:8085/data.json

Рисунок 1.23. Фрагмент вмісту JSON-файлу

Якщо відкрити в браузері посилання, то можна побачити вміст JSON-файлу, який надає локальний веб-сервер Open Hardware Monitor. За цим посиланням ми отримуємо «сирі» моніторингові дані з OHM у форматі JSON.

Вони — основа роботи всього пристрою, бо саме з них Python-скрипт отримує інформацію, яку потім бачимо на OLED-дисплеї.

Цей JSON-файл містить ієрархічну структуру, яка описує всі пристрої, сенсори та їх поточні значення. Python-скрипт за цією адресою отримує JSON-дані, а далі парсить структуру щоб витягнути необхідні параметри:

```
response = requests.get(OHM_URL)                # Отримання JSON-даних з Open Hardware Monitor
data = response.json()                          # Десеріалізація JSON у Python-структуру

cpu_load = find_value(data, 'CPU Total', 'Load')  # Витяг значення навантаження CPU
gpu_load = find_value(data, 'GPU Core', 'Load')   # Витяг значення навантаження GPU
ram_load = find_value(data, 'Memory', 'Load')     # Витяг значення навантаження RAM
```

Рисунок 1.24. Фрагмент коду отримання та обробка JSON-даних з OHM у Python

- CPU Total — загальне навантаження на процесор.
- GPU Core (з розділу Load) — навантаження на відеокарту.
- RAM — використання оперативної пам’яті.

Враховуючи те, що структура JSON може містити декілька сенсорів з однаковими іменами, але різними батьківськими розділами (наприклад, "GPU Core" у розділі "Clocks" і "Load"), було реалізовано функцію, яка шукає потрібне значення за іменем сенсора і його батьківською категорією.

Обробка та форматування даних:

Після того, як отримуються потрібні значення, вони очищуються від символів "%", коми замінюються на крапки, так як числові значення можуть набувати формат 27,5%, а це не відповідає синтаксису float у Python, і дані перетворюються на числа з плаваючою точкою (float), потім вони округлюються до цілих значень (int) для компактного відображення.

```
CPU:24 GPU:9 RAM:35
```

Рисунок 1.25. Приклад форматованого рядка

Цей рядок відповідає формату, який обробляє Arduino.

Надсилання даних через COM-порт:

Рядок, який був відформатований, передається через COM-порт за допомогою бібліотеки PySerial. Зазначена швидкість передачі — 115200 бод, що відповідає налаштуванням скетча Arduino. Кожен рядок завершується символом нового рядка \n, що дозволяє Arduino чітко визначити межу пакету.

Приклад команди відкриття порту:

```
ser = serial.Serial(COM_PORT, BAUD_RATE, timeout=1)
```

Рисунок 1.26. Фрагмент коду команди відкриття порту

Перед кожним надсиланням відбувається невелика пауза:

```
time.sleep(1)
```

Рисунок 1.27. Фрагмент коду виклик функції паузи у виконанні скрипта

Це необхідно для того щоб запобігти перевантаженню послідовного порту і досягти рівномірного оновлення дисплея.

Відлагодження та помилки:

Python-скрипт містить обробку винятків, що запобігає аварійному завершенню програми, як при втраті зв’язку, так і виникненні помилок з доступом до COM-порту.

Якщо відбувається така ситуація, то виконання скрипту не завершується аварійно, а переходить у режим очікування та перезапускає запит.

Якщо знову відбувається помилка чи зв'язок не відновлюється скрипт чекає 5 секунд і повторює спробу.

```
except Exception as e:
    print(f"Помилка: {e}")
    time.sleep(5)
```

Рисунок 1.28. Фрагмент коду Python-скрипту, що забезпечує стійкість роботи програми при втраті зв’язку з COM-портом

Переваги та гнучкість рішення:

Застосування мови програмування Python забезпечує гнучкість у зміні логіки роботи скрипта, можливість швидко адаптувати код до нових параметрів, платформ. Якщо в майбутньому буде потрібно змінити частоту оновлення чи додати інші сенсори, то це можна реалізувати де кількома рядками коду.

Python-скрипт виконує центральну функцію у всій системі: отримує реальні дані про навантаження на комп'ютер і перетворює їх у зручну для Arduino форму. Це дало можливість створити систему моніторингу з відкритим кодом, що не потребує спеціального ПЗ чи драйверів, і легко масштабується чи, за потреби, адаптується під інші задачі.

1.9.2 Розробка скетча Arduino: прийом і обробка даних

Програмна частина, реалізована на базі платформи Arduino Nano, відповідає за отримання форматованих даних з COM-порту, їх парсинг, обробку та передачу на дисплей. Скетч був написаний на мові C++ у середовищі Arduino IDE із використанням бібліотек Wire, Adafruit_GFX, Adafruit_SSD1306. Програма працює у режимі реального часу, що дає можливість реалізувати надійний обмін даними та стабільну взаємодію з Python-скриптом на стороні ПК.

Ініціалізація компонентів

Після запуску мікроконтролера, виконується ініціалізація серійного послідовного з'єднання та OLED-дисплея. Це відбувається у функції setup():

```
Serial.begin(115200);
delay(1000);

if (!display.begin(SSD1306_SWITCHCAPVCC, OLED_ADDRESS)) {
  Serial.println(F("SSD1306 allocation failed"));
  while (true); // Стоп, якщо дисплей не ініціалізувався
}
```

Рисунок 1.29. Фрагмент коду ініціалізації послідовного з'єднання та перевірки OLED-дисплея SSD1306

Дисплей очищується і на ньому з’являється стартове повідомлення

```
display.clearDisplay();
display.setTextSize(2);
display.setTextColor(SSD1306_WHITE);
display.setCursor(0, 0);
display.println(F("Waiting..."));
display.display();
```

Рисунок 1.30. Фрагмент коду виводу початкового повідомлення на OLED-дисплей перед отриманням даних

Прийом даних із COM-порту

Мікроконтролер постійно прослуховує COM-порт символ за символом.

```
while (Serial.available()) {
  char inChar = (char)Serial.read();
  if (inChar == '\n') {
    processData(inputString);
    inputString = "";
  } else {
    inputString += inChar;
  }
}
```

Рисунок 1.31. Фрагмент коду зчитування серійних даних і передача їх на обробку після завершення рядка

Парсинг та обробка рядка

Надісланий рядок має фіксовану структуру:

```
CPU:24 GPU:9 RAM:35
```

Рисунок 1.32. Приклад фіксованої структури рядка

Функція processData() використовує метод indexOf() для пошуку ключових міток та substring() для виділення значень:

```
int cpuStart = data.indexOf("CPU:");
int gpuStart = data.indexOf("GPU:");
int ramStart = data.indexOf("RAM:");

String cpu = data.substring(cpuStart + 4, gpuStart);
String gpu = data.substring(gpuStart + 4, ramStart);
String ram = data.substring(ramStart + 4);
```

Рисунок 1.33. Фрагмент коду виділення значень CPU, GPU та RAM із отриманого рядка даних

Кожне з цих значень обрізається методом trim() та виводиться на дисплей.

```
cpu.trim(); gpu.trim(); ram.trim();
```

Рисунок 1.34. Фрагмент коду очищення отриманих значень CPU, GPU та RAM від зайвих пробілів

Вивід на екран

Щоб відображувати інформацію на OLED-дисплеї використовується функція display.println(), вона є частиною бібліотеки Adafruit_SSD1306. Для встановлення розміру шрифту використовується команда display.setTextSize(...), якщо не вказати розмір шрифту у полі "..." то за замовчуванням буде використано розмір 1. У даній реалізації використовується розмір 2 для кращої видимості.

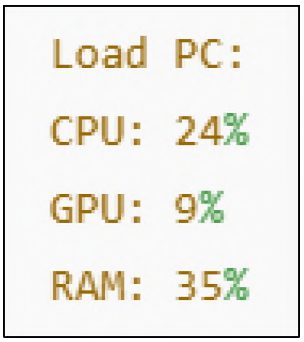


Рисунок 1.35. Приклад даних, які виводяться на дисплей

Надійність та помилки

Скетч передбачає базову обробку помилок, зокрема перевірку на наявність підрядків у вхідному рядку. Якщо формат невірний — на дисплей виводиться повідомлення:

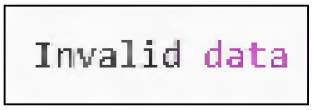


Рисунок 1.36. Приклад повідомлення про помилку у вхідних даних, які виводяться на дисплей

Після виведення повідомлення про помилку у вхідних даних система не припиняє свою роботу, а повертається до циклу очікування нового рядка даних через COM-порт. Після їх отримання виконується повторна перевірка формату. Якщо дані коректні — інформація буде виведена на дисплей. Якщо ж знову буде зафіксовано помилку — повідомлення про помилку у вхідних даних з'явиться знову. Такий підхід гарантує безперервність роботи пристрою навіть у випадку непередбачуваних збоїв чи втрати зв'язку.

Можливість масштабування

Алгоритм обробки легко масштабувати, додаючи нові ключі у рядок даних. Наприклад, можна додати TEMP:, SSD:, FPS:, NET: використавши: display.setCursor(...) та display.print(...) для виведення нових параметрів на окремих рядках дисплея.

Скетч Arduino реалізує ефективну, легку для розуміння логіку прийому даних, їх обробки та виводу, забезпечує стабільну роботу у зв'язці з Python, не потребує складних структур керування, і може бути розширений без кардинальної зміни архітектури.

1.9.3 Форматування та вивід даних на дисплей

Відображення інформації на OLED-дисплею є фінальною фазою в алгоритмі роботи пристрою. Саме на цьому етапі користувач отримує візуалізацію поточного стану ПК за обраними параметрами.

У проєкті використовується OLED-дисплей з роздільною здатністю 128x64 пікселі побудований на базі контролера SSD1306, що підключається через I^2C-шину що включає лінії SDA (дані) та SCL (синхронізація). Стандартна I^2C-адреса дисплея — 0x3C, що є типовим для більшості OLED-модулів. Адреса задається апаратно або фіксовано модулем.

Ініціалізація дисплея

За допомогою функції setup() відбувається перевірка доступності пристрою за I^2C.

```
void setup() {
  Serial.begin(115200);
```

Рисунок 1.37. Фрагмент коду ініціалізації OLED-дисплея SSD1306 із з перевіркою доступності пристрою по I2C

Якщо дисплей не виявлено чи відсутнє з'єднання— виконується зупинка пристрою циклом while (true);, і, через Serial, передається повідомлення про помилку:

```
if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
  Serial.println(F("Error: SSD1306 display not found"));
  while (true);
}
```

Рисунок 1.38. Фрагмент коду перевірки доступності OLED-дисплея та повідомлення про помилку

Структура виводу

Відображення даних на дисплеї розділено на 4 логічні рядки:

1. Заголовок: Load PC:
2. CPU навантаження: CPU: 24%
3. GPU навантаження: GPU: 9%
4. RAM використання: RAM: 35%

Кожна з позицій розміщується із заданим відступом за координатами setCursor(x, y), а для зручності використовується шрифт 2-го розміру:

```
display.setTextSize(2);
display.setCursor(0, 0);
display.println("Load PC:");
...
```

Рисунок 1.39. Фрагмент коду встановлення шрифту, координат і виведення заголовка на OLED-дисплей

Очищення та оновлення

Перед кожним новим виводом екран очищується командою:

```
display.clearDisplay();
```

Рисунок 1.40. Фрагмент коду очищення екрану OLED-дисплея перед оновленням вмісту

Після відображення усіх рядків виконується:

```
display.display();
```

Рисунок 1.41. Фрагмент коду оновлення зображення на OLED-дисплеї після відображення усіх рядків

Ця функція фізично оновлює буфер дисплея.

Енергоспоживання:

Оскільки, дисплей використовує технологію OLED (пікселі споживають енергію лише при виведенні світла), то вивід даних не створює значного навантаження на енергобаланс пристрою, а це важливо для проєктів із живленням через USB чи батарею.

Можливість розширення структури виводу:

Можна змінити розмір шрифтів, додати інші вивідні елементи, іконки та інше. Також можна: застосувати анімацію, змінити структуру даних на графіки чи гістограми навантаження, також можна прокручуваний текст.

Після реалізації механізму виведення системних параметрів на дисплей, необхідно здійснити перевірку коректності передачі даних між Python-скриптом та мікроконтролером, відповідність формату вхідного рядка очікуваній структурі, а також стабільність відображення параметрів на OLED-дисплеї.

Для з’єднання компонентів були використані кабелі Dupont типу "мама-мама". На початковому етапі збирання було використано чотири кабелі й підключили OLED-дисплей до мікроконтролера Arduino Nano відповідно до схеми, що зображена на рисунку 1.20.

Після підключення спостерігаємо, що дисплей частково засвічений, проте видно "сміття" — хаотичні пікселі замість коректного виводу тексту чи графіки.

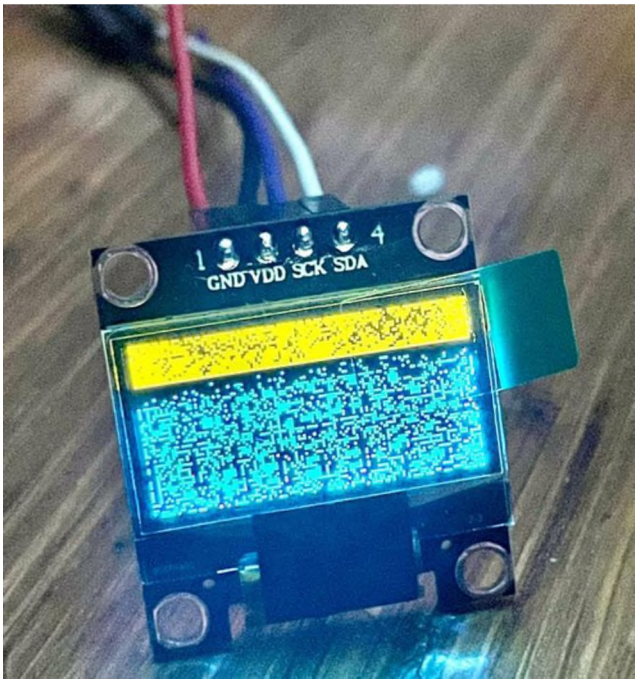


Рисунок 1.42. Фото несправного виводу на OLED-дисплей SSD1306

Після діагностики з'єднань було виявлено, що причиною некоректного відображення даних на OLED-дисплеї був поганий контакт SDA кабеля, що призводило до виведення "зашумленого" зображення. Після перепідключення та надійного фіксування контактів у з’єднанні — дисплей почав стабільно відображати необхідну інформацію про стан CPU та GPU. Значення тепер виводяться у заданому форматі, що підтверджує правильність роботи як програмної, так і апаратної частини проєкту (рисунок 1.43.).

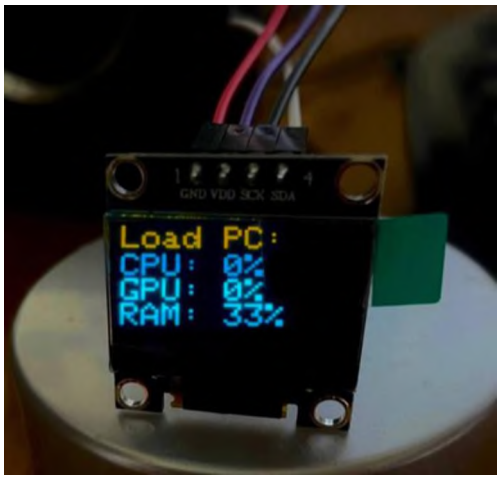


Рисунок 1.43. Фото результату коректної роботи дисплея

Реалізований механізм виводу даних на OLED-дисплей дозволяє зручно й наочно відображати діагностичну інформацію в реальному часі. Структура виводу

легко адаптується до нових вимог і може бути легко, як змінена, так розширена для відображення додаткових параметрів чи візуальних елементів.

1.10 Збирання апаратної частини пристрою

Після завершення програмної реалізації та вибору компонентної бази було здійснено фізичне збирання апаратної частини пристрою.

До складу пристрою увійшли:

– Arduino Nano, як центральний обчислювальний модуль, який обробляє отримані дані та керує відображенням;

– OLED-дисплей SSD1306 із роздільною здатністю 128×64 пікселі, який використовується для виведення інформації про навантаження системи;

– USB-кабель типу mini-B — для забезпечення живлення мікроконтролера та передавання даних між ПК і пристроєм;

– набір кабелів Dupont — для зручного монтажу та комутації елементів без потреби пайки.

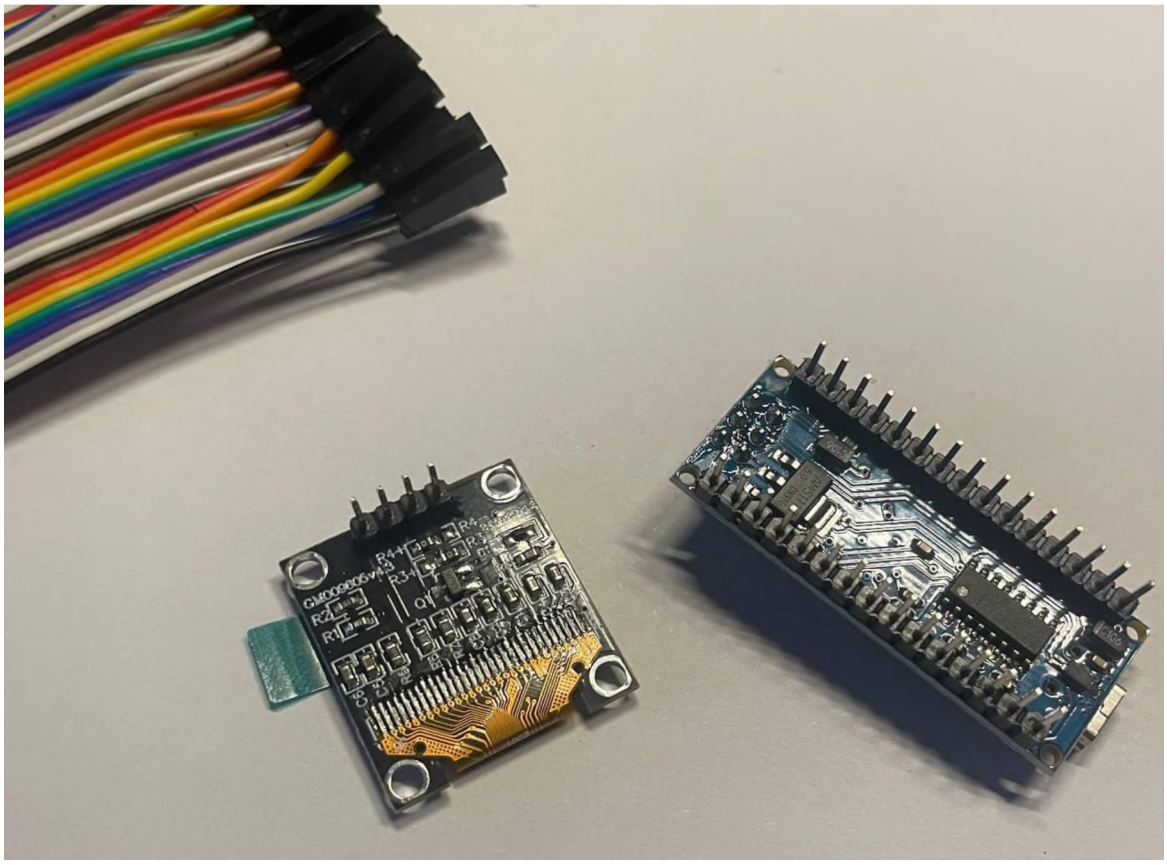


Рисунок 1.44. Фото основних компонентів проєкту до монтажу

На рисунку 1.44. зображено всі основні компоненти, які використовуватимуться під час складання пристрою моніторингу стану ПК. Корпус пристрою планується виготовити вручну за допомогою 3D-ручки, яка працює на основі розплавлення полімерної нитки типу PLA. Це дозволить гнучко формувати індивідуальну форму для надійного розміщення всіх компонентів пристрою.

Починаємо монтування апаратної частини пристрою.

Встановлено мікроконтролер Arduino Nano, зафіксований попередньо підготовленому місці у саморобному корпусі. USB-роз'єм розміщено зліва для підключення до ПК, праворуч — OLED-дисплей SSD1306, закріплений у корпусі

Це надійно фіксує плату та дисплей, забезпечуючи їх захист від механічних пошкоджень (рисунок 1.45.).

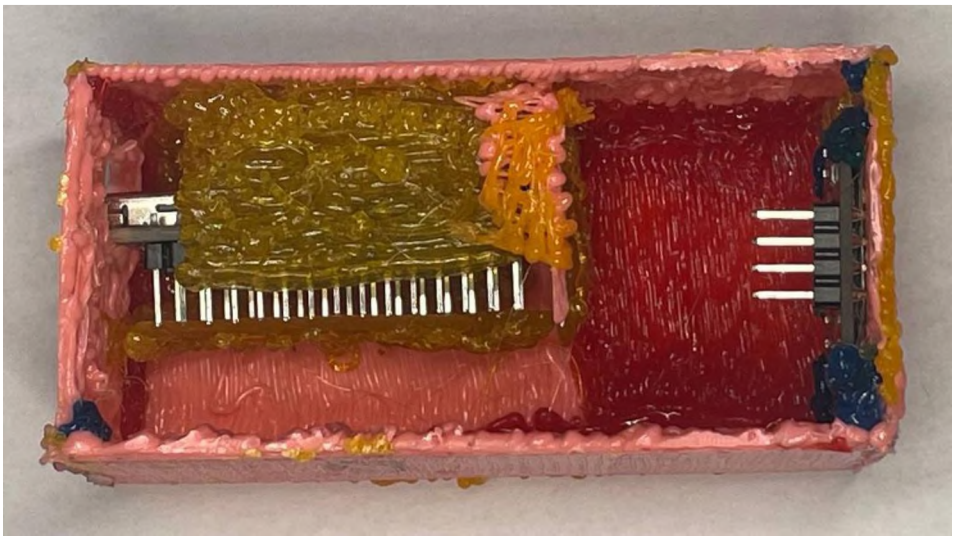


Рисунок 1.45. Фото початкового етапу монтування корпусу пристрою моніторингу на базі Arduino Nano

Завершуємо компоновку елементів усередині корпусу пристрою моніторингу: з'єднуємо всі необхідні кабелі з відповідними пін-модулями — видно дроти живлення, даних (SDA, SCL) та землі, які йдуть до OLED-дисплея.

Праворуч — OLED SSD1306, закріплений у корпусі, а кабелі під’єднано за допомогою Dupont-конекторів (рисунок 1.46.).

Цей монтаж дозволяє:

– уникнути коротких замикань;

– забезпечити захист плати від механічного пошкодження;

– зручно розмістити всі елементи в компактному корпусі.

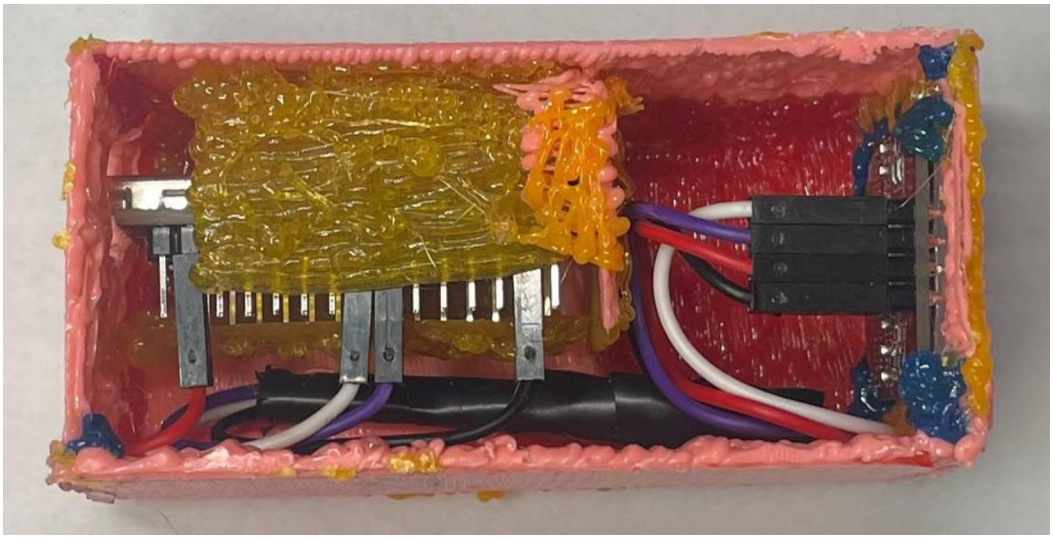


Рисунок 1.46. Зображення завершеного внутрішнього з'єднання компонентів у корпусі пристрою

Потім було проведено фінальне складання. Корпус надійно скріплено по швах та щільно закрито. Підключення USB-кабелю здійснюється з тильної сторони — це забезпечує живлення Arduino Nano і передачу даних із ПК. Пристрій має компактний вигляд і є зручним для використання на робочому столі поряд із ПК.

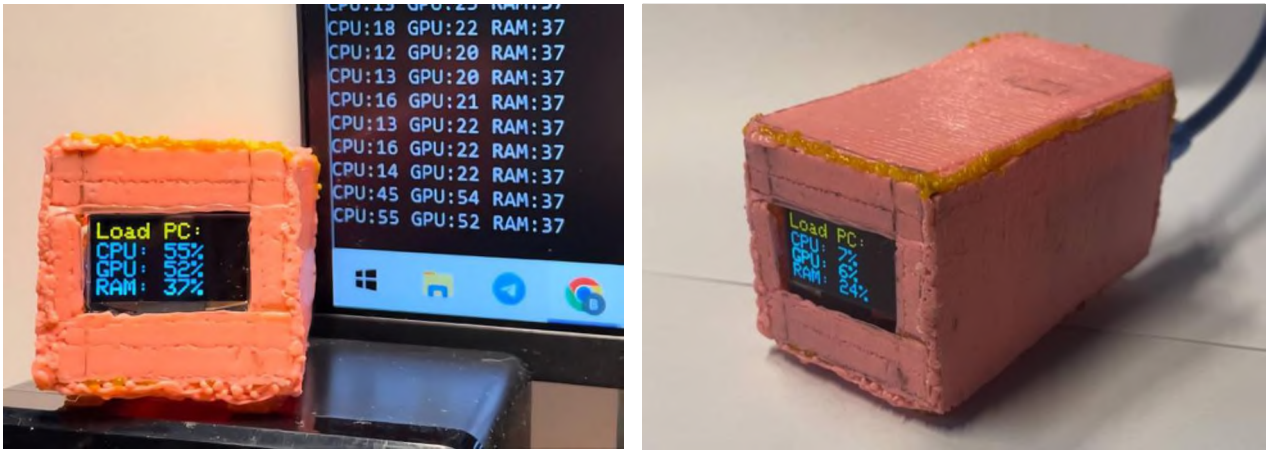


Рисунок 1.47. Зображення підключеного до ПК пристрою

1.11 Тестування роботи пристрою

Після завершення підбору апаратних компоненти та розробки програмної частини пристрою, було проведено тестування його роботи у реальних умовах. У цьому розділі буде описано тестування роботи пристрою для моніторингу стану ПК.

Основна мета: перевірка стабільності зв’язку, коректність обробки вхідних даних та виводу інформації в реальному часі.

Етапи тестування проілюстровані на рисунку 1.48.

Рисунок 1.48. Схема тестування приладу

Тестування проводилося у кілька етапів:

1.Перевірка ініціалізації компонентів.

Починається ініціалізація OLED-дисплея через I^2C-шину. Дисплей показує заставку:

Рисунок 1.49. Зображення екрану

Це говорить про те що, пристрій готовий до прийому даних.

У випадку, якщо під час ініціалізації виникає помилка, наприклад, через помилку з'єднання, неправильну I²C-адресу, виводиться відповідне повідомлення у моніторі порту Arduino IDE:

```
SSD1306 allocation failed
```

Рисунок 1.50. Приклад повідомлення у моніторі порту Arduino IDE

2. Запуск програми Open Hardware Monitor на комп'ютері

Запускаємо програму Open Hardware Monitor, яка відкриває доступ до http://localhost:8085/data.json

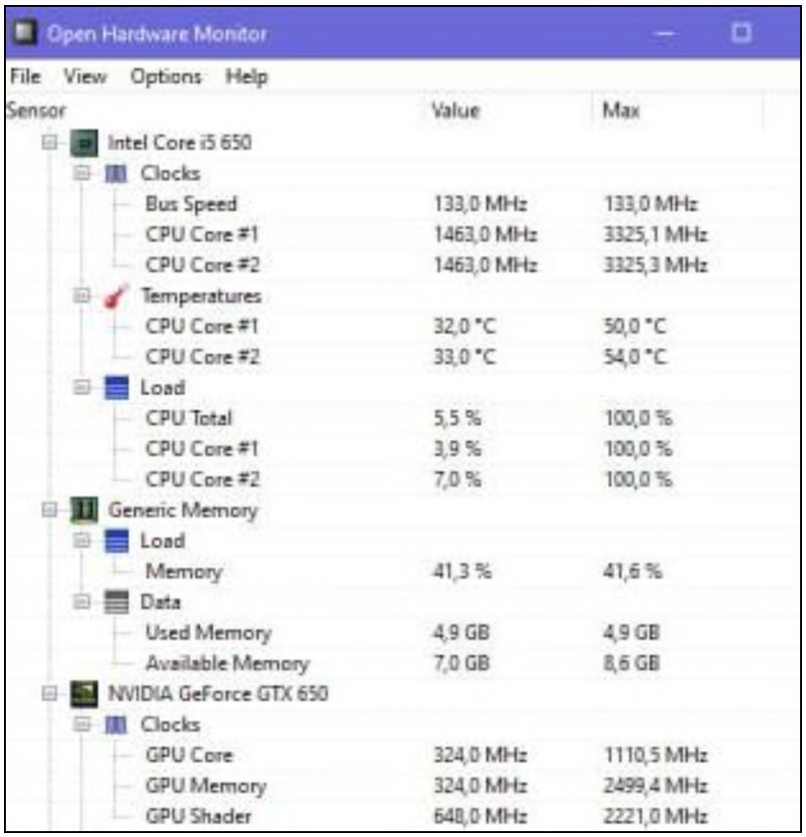


Рисунок 1.51. Загальний вигляд вікна програми

3.Запуск Python-скрипта на комп'ютері

Відкриваємо файл ohm_to_arduino.py, у якому реалізовано основну логіку зчитування даних із Open Hardware Monitor. Скрипт звертається до локального веб-інтерфейсу за адресою http://localhost:8085/data.json.

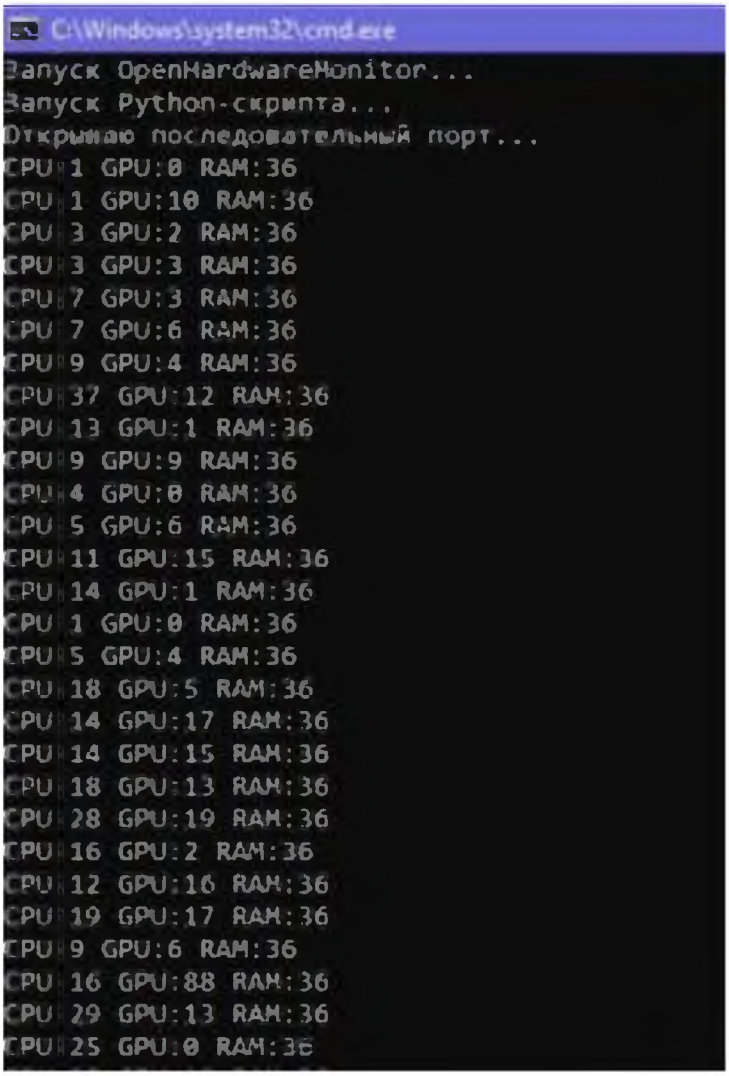


Рисунок 1.52. Загальний вигляд вікна програми

Скрипт обробляє JSON-відповідь, формує рядок у форматі Python-скрипт успішно встановлює зв'язок з веб-сервером, аналізує JSON-структуру, виділяє потрібні параметри та формує рядок:

```
CPU:24 GPU:9 RAM:35
```

Рисунок 1.53. Приклад фіксованої структури рядка

Потім надсилає його у COM-порт (наприклад, COM8) зі швидкістю 115200 бод.

4. Автоматизація запуску

Для зручності використання було створено BAT-файл, який називається: «запуск_монітора.bat». він автоматично запускає всі необхідні програми для роботи проєкту, що оптимізує запуск програмної частини приладу до одного

подвійного кліку:

```
@echo off
chcp 65001 >nul

echo Запуск OpenHardwareMonitor...
start "" "C:\Users\user\Desktop\OpenHardwareMonitor\OpenHardwareMonitor.exe"

timeout /t 5 >nul

echo Запуск Python-скрипта...
cd /d "C:\Users\user\Desktop"
python ohm_to_arduino.py

pause
```

Рисунок 1.54. Код BAT-файлу

5. Оцінка оновлення дисплея та читабельності інформації

Під час тестування пристрою значну увагу було приділено якості виведення інформації на OLED-дисплей. Це дуже важливо, адже основною функцією пристрою є надання користувачеві актуальних і чітко візуалізованих даних про навантаження системи.

Оцінка проводилась за декількома критеріями:

Чіткість відображення:

Значення параметрів (CPU, GPU, RAM) відображаються чітко та розбірливо. Текст виводиться великим шрифтом, це дозволяє легко сприймати інформацію і не потрібно вдивлятися в екран чи напружувати зір.

Читабельність з робочої відстані:

Оцінка проводилась на відстані 30–50 см від користувача, це типова відстань для розміщення настільної електроніки.

Інформація залишається цілком доступною для зорового сприйняття. Кожен рядок (CPU, GPU, RAM) виводиться окремо з достатнім міжрядковим інтервалом.

6. Перевірка синхронізації та затримки виводу

Для забезпечення коректної роботи системи моніторингу важливо перевірити синхронізацію між усіма компонентами та оцінити затримку при передачі даних. Надійність взаємодії між ПК, Python-скриптом, Arduino та OLED-

дисплеєм безпосередньо впливає на точність та швидкість виводу інформації.

Швидкість оновлення:

Завдяки циклічному опитуванню Open Hardware Monitor кожну секунду та миттєвій передачі цих даних через COM-порт, дисплей оновлює значення з мінімальною затримкою. Що забезпечує майже реальний час оновлення показників.

Візуалізація даних:

Під час тривалого використання дисплея не спостерігалося мерехтіння чи затримок у виведенні даних, що робить спостереження комфортними. Оновлення дисплея відбувається плавно.

Відсутність "залипання" попередніх значень — кожне нове оновлення повністю стирає попередній вивід через команду clearDisplay().

7. Оцінка роботи скрипта Python при зміні навантаження на ПК.

Для перевірки ефективності роботи Python-скрипта тестування було проведено на комп'ютері з процесором Intel Core i5 650, відеокартою Nvidia GeForce GTX 650 (1 ГБ) та 12 ГБ оперативної пам'яті DDR3. Основна мета тестів — оцінити швидкість збору та обробки даних, стабільність передачі через COM-порт та відповідність фактичних значень зміні навантаження на ПК.

Для імітації зміни стану ПК застосовано:

– Відкриття кількох вкладок у браузері Google Chrome.

Рисунок 1.55. Навантаження на ПК під час відкриття кількох вкладок у браузері

Запуск спричиняє зростання CPU: до 60–70%;

– Відтворення відео у форматі FullHD.

Рисунок 1.56. Навантаження на ПК під час відтворення відео FullHD

Запуск спричиняє зростання CPU: до 80–90% та GPU: до 65–75%;

– Тестування під час роботи гри Metro 2033 Redux на середніх графічних налаштуваннях.

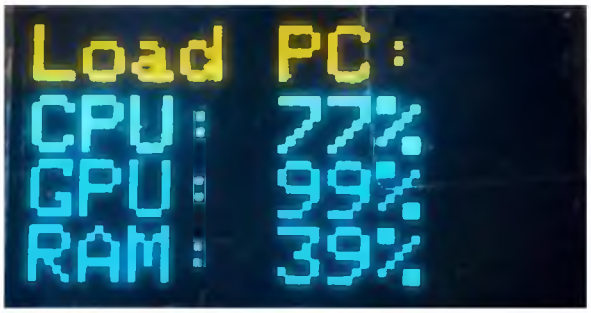


Рисунок 1.57. Навантаження на ПК під час роботи гри

Запуск спричиняє зростання GPU: до 85–99%

Тестування показало, що скрипт ефективно збирає, обробляє та передає дані про стан ПК навіть за умов високого навантаження.

8.Визначення стабільності при тривалому увімкненні

Тестування тривалої роботи проводилось упродовж 4 годин безперервної роботи пристрою.

Результати:

Arduino Nano стабільно приймає та обробляє дані без збоїв;

OLED-дисплей не втрачає яскравості, відсутні артефакти зображення.

Переваги створеного пристрою:

– працює без встановлення додаткового ПЗ на стороні Arduino;

– працює без інтернет-з’єднання чи зовнішніх бібліотек, окрім Adafruit_SSD1306 та Wire;

– проста адаптація під інші параметри (можна додати температуру CPU,

FPS, використання SSD тощо);

– реальна автономність: після запуску скрипта пристрій працює без участі користувача — просто відображає дані у зручному форматі на дисплеї.

За результатами тестування можна зробити наступні висновки:

– пристрій збирає та передає дані. Зміни навантаження відображаються і змінами значень CPU, GPU, RAM;

– за період спостереження не було втрати з'єднання з Arduino.

– відображення даних на OLED-дисплеї чітке, легко читається з відстані 30–50 см, без артефактів;

– оновлення даних з інтервалом в 1 секунду достатньо для моніторингу в реальному часі;

– пристрій придатний до щоденного використання і не потребує додаткового налаштування чи ручного втручання;

– низьке енергоспоживання дозволяє живити пристрій безпосередньо від USB-порту ПК.

Тестування підтвердило працездатність, стабільність і практичну користь пристрою. Він може застосовуватися як елемент персонального моніторингу ПК чи демонстраційний прилад. А також проєкт має потенціал для подальшого розвитку і масштабування, зокрема у напрямку бездротової передачі даних, а також можливе створення версії з веб-інтерфейсом.

1.12 Інструкція з адаптації для різних комп'ютерів

У цьому розділі буде розміщено докладну інструкцію, яка допоможе кожному бажаючому налаштувати пристрій для діагностики стану ПК на будь-якому іншому комп'ютері. Оскільки пристрій був спеціально спроєктований із врахуванням портативності для його повноцінної роботи потрібно лише декілька простих кроків. Це буде корисно у випадках, коли пристрій демонструється на іншому обладнанні, наприклад, під час презентацій у навчальних закладах.

Загальні вимоги до середовища

Для коректної роботи пристрою потрібно:

Комп’ютер під управлінням Windows 7/10/11.

Наявність USB-порту для підключення Arduino Nano.

Підключення до мережі Інтернет (на етапі встановлення бібліотек).

1. Підготовка програмного забезпечення

– Завантажте та запустіть Open Hardware Monitor (OHM):

– Перейдіть на сайт: https://openhardwaremonitor.org/

– Завантажте ZIP-архів, розпакуйте його в зручне місце.

– Запустіть файл OpenHardwareMonitor.exe.

– Ввімкніть пункт Options → Remote Web Server → Run, щоб активувати локальний сервер:

Доступні дані за адресою: http://localhost:8085/data.json.

– Встановіть Python (версія 3.10 і вище):

– Завантажте: https://www.python.org/downloads/windows/

Під час встановлення обов’язково поставте галочку Add Python to PATH.

– Перевірте встановлення через командний рядок:

```
python --version
pip --version
```

Рисунок 1.58. Перевірка версій Python і pip у командному рядку.

– Встановіть необхідні бібліотеки: відкрийте командний рядок та введіть:

```
>pip install requests pyserial
```

2. Підключення пристрою:

– Підключіть Arduino Nano до ПК через USB-інтерфейс.

– Дочекайтеся автоматичної установки драйверів.

Якщо драйвер не встановлюється автоматично, завантажте CH340/FTDI драйвер із офіційного джерела (залежно від моделі).

– Визначте COM-порт, до якого підключено Arduino:

Натисніть Win+X → Диспетчер пристроїв → Порти (COM та LPT) → Arduino буде відображено як COMX(наприклад 'COM3').

– Відредагуйте Python-скрипт:

Відкрийте ohm_to_arduino.py у будь-якому текстовому редакторі (наприклад, Notepad++).

Змініть рядок: COM_PORT = 'COM5'

на актуальний COM-порт вашого Arduino (наприклад, 'COM3').

Збережіть зміни.

– Запустіть програму Open Hardware Monitor, яка відкриває доступ до http://localhost:8085/data.json

– Запустіть Python-скрипт на комп'ютері: відкрийте файл ohm_to_arduino.py, у якому реалізовано зчитування даних із Open Hardware Monitor через http://localhost:8085/data.json.

У якості альтернативи.

Запустіть BAT-файл, який називається: «запуск_монітора.bat» і запускає програмну частину приладу (Open Hardware Monitor та Python-скрипт) одним подвійним кліком, якщо обидва файли знаходяться на робочому столі:

```
@echo off
chcp 65001 >nul

echo Запуск OpenHardwareMonitor...
start "" "C:\Users\user\Desktop\OpenHardwareMonitor\OpenHardwareMonitor.exe"

timeout /t 5 >nul

echo Запуск Python-скрипта...
cd /d "C:\Users\user\Desktop"
python ohm_to_arduino.py

pause
```

Рисунок 1.59. Код BAT-файлу

За потреби, Ви можете змінити розташування цих файлів, але потрібно буде змінити шлях до програм у самому BAT- файлі.

3.Перевірка роботи пристрою

Якщо після запуску скрипта на екрані OLED з'явиться напис (див. рисунок 1.60.) то Ви зробили все правильно і прилад працює коректно.

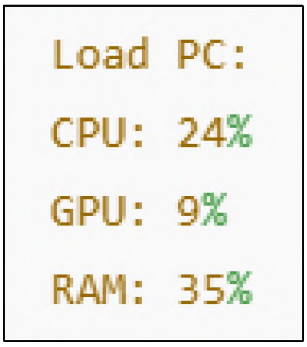


Рисунок 1.60. Приклад даних, які виводяться на дисплей

Цей пристрій є універсальним і не прив’язаний до конкретної конфігурації ПК, за умови встановленого OHM, Python та правильного COM-порту. Процес встановлення займає приблизно 10 хвилин навіть на незнайомій машині.

2 ЕКОНОМІЧНИЙ РОЗДІЛ

У цьому розділі будуть зроблені розрахунки вартості виконання науково-дослідної роботи «Розробка пристрою моніторингу навантаження комп'ютера з виведенням даних на OLED-дисплей». Розроблений апаратно-програмний комплекс призначений для відображення в реальному часі інформації про завантаження основних компонентів комп'ютера (центрального процесора, графічного процесора та оперативної пам'яті) з використанням зовнішнього мікроконтролера Arduino та компактного дисплея.

Було розроблено проєкт відноситься до категорії науково-дослідницьких розробок (далі НДР) у галузі прикладної комп'ютерної інженерії. Оцінка ефективності запропонованого рішення включає розрахунок трудомісткості етапів розробки та загальної вартості реалізації проєкту.

Розрахунок трудомісткості НДР здійснювався у такій послідовності:

1) Я склав перелік основних етапів і видів робіт, що необхідно буде виконати в процесі реалізації цього проєкту. Після обговорення з науковим керівником були уточнені назви етапів, об'єднані та виключені окремі види робіт, що не мають істотного впливу на результат.

2) Для кожного етапу було визначено необхідний рівень кваліфікації виконавця. Розподіл робіт по етапах і виконавцях наведено у таблиці 2.1.

Таблиця 2.1. Розподіл робіт по етапах і видах виконавців

№ п	Етап проведення	Вид роботи	Посада виконавця
1	Розробка технічного завдання (ТЗ)	Складання і затвердження ТЗ для НДР по розробці «Розробка пристрою діагностики стану ПК на платформі Arduino».	Дипломник, керівник
2	Вибір напряму дослідження	Збір і вивчення науково-технічної літератури, технічної документації і інших матеріалів, на основі яких будуватиметься робота.	Дипломник, керівник

		Формулювання можливих напрямів вирішення завдань, поставлених в технічному завданні НДР і їх порівняльна оцінка.	
		Вибір напряму проведення досліджень для подальшої розробки.	
		Розробка плану проведення досліджень для подальшої розробки.	
		Огляд існуючих комерційних рішень для контролю стану ПК.	
		Визначення основних параметрів моніторингу.	
3	Теоретичні і експериментальні дослідження	Розробка моделі пристрою і визначення базових параметрів.	Дипломник, керівник, консультанти
		Підбір компонентів пристрою та обгрунтування вибору.	
		Розробка схеми підключення елементів пристрою.	
		Реалізація програмного забезпечення.	
		Збирання апаратної частини пристрою.	
		Тестування роботи пристрою.	
		Узагальнення результатів попередніх етапів роботи.	
4	Узагальнення і оцінка результатів досліджень	Оцінка повноти вирішення поставлених завдань.	Дипломник, керівник, консультанти
		Складання і оформлення звіту. Розгляд результатів проведеною НДР і прийняття результатів в цілому.	

Час виконання окремих завдань в умовах відсутності нормативної бази тривалість виконання окремих робіт визначається на основі ймовірних оцінок робіт, що надають безпосередньо виконавцями. Це відбувається через те, що відсутні встановлені нормативи часу виконання робіт. Очікувана трудомісткість робіт розрахована у табл. 2.2.

Таблиця 2.2. Очікувана трудомісткість робіт

№ п/п	Вид роботи	Очікуваний час виконання (дні)
1	Складання і затвердження ТЗ для НДР по розробці «Розробка пристрою діагностики стану ПК на платформі Arduino».	1
2	Збір і вивчення науково-технічної літератури, технічної документації і інших матеріалів, на основі яких будуватиметься робота.	2
3	Формулювання можливих напрямів вирішення завдань, поставлених в технічному завданні НДР і їх порівняльна оцінка.	2
4	Вибір напряму проведення досліджень для подальшої розробки.	1
5	Розробка плану проведення досліджень для подальшої розробки.	1
6	Огляд існуючих комерційних рішень для контролю стану ПК.	1
7	Визначення основних параметрів моніторингу	1
8	Розробка моделі пристрою і визначення базових параметрів.	2
9	Підбір компонентів пристрою та обґрунтування вибору.	2
10	Розробка схеми підключення елементів пристрою	1
11	Реалізація програмного забезпечення	3
12	Збирання апаратної частини пристрою.	
13	Тестування роботи пристрою	2
14	Узагальнення результатів попередніх етапів роботи.	3
15	Оцінка повноти вирішення поставлених завдань.	2
16	Складання і оформлення звіту. Розгляд результатів проведеною НДР і прийняття результатів в цілому.	3
	Разом	27

Розрахунок собівартості і ціни виконання НДР:

– Матеріальні витрати:

Витрати на матеріали складають:

200 грн. (Папір А4 1 пачка на 500 листів) = 200 грн.

– Заробітна плата — оплата праці виконавців, які безпосередньо приймають участь.

Розмір основної зарплати встановлюється виходячи з чисельності різних категорій виконавців, трудомісткості, що витрачається ними на виконання різних видів робіт, а також їх середньої заробітної плати (ставки) за один робочий день. Відповідно до статті 8 «Закону про Державний бюджет України на 2025» встановлено мінімальну заробітну плату у місячному розмірі з 1 січня 2025 року — 8000 гривень; мінімальну погодинну тарифну ставку – 48,00 грн.

Середня зарплата за один робочий день для кожного виконавця визначена по формулі:

$$З_{ден} = п.т.с. * 8 \quad (2.1)$$

де п.т.с - погодинна тарифна ставка, грн.;

8 — тривалість робочого дня, год.

$З_{ден}$ дипломника = 48,00*8 = 384,00 грн.

$З_{ден}$ керівника = 72,00 *8 = 576 грн.

$З_{ден}$ консультантів = 72,00 *8 = 576 грн.

Розрахунки заробітної плати відображено в Таблиця 2.3.

Таблиця 2.3. Витрати на основну заробітну плату

Виконавець	Погодинна тарифна ставка, грн.	Денна ставка, грн.	Трудомісткість робочих днів	Сума основної зарплати, грн.
Дипломник	48,00	384,00	27,00	10 368,00
Керівник	72,00	576,00	1,00	576,00
Консультант поекономічній частині	72,00	576,00	0,25	144,00
Консультант по охороні праці	72,00	576,00	0,25	144,00
Нормоконтроль	72,00	576,00	0,25	144,00
Всього (Зо)				11 376,00

Розмір на додаткової заробітної плати визначаються у відсотках від основної Розмір додаткової заробітної плати визначається у відсотках від основної заробітної плати і може включати доплати, надбавки, гарантійні та компенсаційні виплати, передбачені законодавством, а також премії. У наукових закладах додаткова заробітна плата складає 10-100% від основної заробітної плати.

$$З_д = 10\%З_о \quad (2.2)$$

$З_д = 11\ 376{,}00*0{,}10 = 1\ 137{,}6$ грн

– Сума Єдиного соціального внеску на загальнообов'язкове державне соціальне страхування складає 22%:

$$З_{єсв} = 0{,}22*(Зо+Зд) \quad (2.3)$$

$З_{єсв} = 0{,}22*(11\ 376{,}00+1\ 137{,}6) = 2\ 752{,}99$ грн.

– Накладні витрати — це витрати на управління і господарське обслуговування, що відноситься до всіх виконуваних НДР. У наукових закладах накладні витрати складають 40-120% від основної і додаткової заробітної плати.

$$Р_{накл} = (З0+Зд)*0{,}5 \quad (2.4)$$

$Р_{накл} = (11\ 376{,}00+1\ 137{,}6)*\ 0{,}5 = 6\ 256{,}80$ грн.

Тепер розрахую калькуляцію планової собівартості в цілому НДР:

Таблиця 2.4. Калькуляція планової собівартості

	Статті витрат	Сума, грн.
1	Матеріали	200,00
2	Основна заробітна плата	11 376,00
3	Додаткова заробітна плата	1 137,60
4	Відрахування до єдиного соціального внеску	2 752,99
5	Накладні витрати	6 256,80
	Планова собівартість ($С_{пл}$)	21 723,39

Плановий прибуток визначений по формулі:

$$П_{пл} = 0,1*С_{пл} \quad (2.5)$$

$П_{пл}$ = 0,1*21 723,39 = 2 172,34 грн.

де 0,1 — норматив, який враховує граничний рівень рентабельності, встановлений чинним законодавством для науково-технічної продукції.

Договірна ціна визначається по формулі:

$$Ц_{нір} = С_{пл} + П_{пл} \quad (2.6)$$

$Ц_{нір}$ =21 723,39 +2 172,34 = 23 895,73 грн.

Звідси ціна реалізації становить:

$$Ц_{р} = Ц_{нір} + ПДВ \quad (2.7)$$

$Ц_{р}$ = 23 895,73+23 895,73*0,2= 28 674,88 грн.

Отже, проєкт є економічно обґрунтованим, а його реалізація забезпечує оптимальне співвідношення функціональності, вартості та простоти впровадження для індивідуального чи малосерійного використання.

3 РОЗДІЛ ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

Основним нормативно-правовим актом, що регулює відносини у сфері охорони праці, є Закон України "Про охорону праці" : «Цей Закон визначає основні положення щодо реалізації конституційного права працівників на охорону їх життя і здоров'я у процесі трудової діяльності, на належні, безпечні і здорові умови праці, регулює за участю відповідних органів державної влади відносини між роботодавцем і працівником з питань безпеки, гігієни праці та виробничого середовища і встановлює єдиний порядок організації охорони праці в Україні» Дотримання його положень є обов'язковим для всіх учасників процесу.

Виконання вимог з питань охорони праці та техніки безпеки є обов’язковим під час розробки пристрою.

3.1 Вимоги до організації робочого місця працівника та ергономіка

Основні вимоги до приміщення: мінімальна площа на одну особу має бути 6 м², що гарантує достатній простір для продуктивної роботи; об’єм приміщення не менше 20 м³, що сприяє належній циркуляції повітря та зменшує ризик накопичення шкідливих речовин; висота приміщення становить 2.5 м та більше, що дозволяє забезпечити оптимальні умови освітлення та розміщення обладнання; вентиляційна система включає природну та примусову вентиляцію, що гарантує стабільний рівень повітрообміну, видалення зайвого тепла від електронної техніки та підтримку комфортного мікроклімату.

Основні вимоги до робочого середовища: освітленість – 300–500 лк (змішане освітлення: природне + штучне); мікроклімат – температура 20–24°C, вологість 40–60%, забезпечені системою вентиляції та кондиціонування; шум – рівень ≤ 50 дБ, що відповідає нормам для комфортної розумової діяльності; електромагнітне випромінювання – рівень OLED-дисплея та Arduino не перевищує допустимих норм, що гарантує безпеку для користувача.

Основні вимоги до робочого місця:

Освітлення робочого місця: оптимальне освітлення сприяє зменшенню напруги очей, тому робоче місце повинно мати достатнє, рівномірне та несліпуче

освітлення. При цьому необхідно уникати прямих сонячних променів і відблисків від освітлювальних приладів на екрані монітора ПК та на дисплеї пристрою моніторингу.

Розташування монітора ПК: Монітор ПК повинен бути розташований на відстані 60-70 см від очей користувача. Верхній край екрана має знаходитись на рівні очей чи трохи нижче, щоб уникнути надмірного нахилу голови, під кутом 90-100 градусів до вікон, забезпечуючи бокове падіння світла.

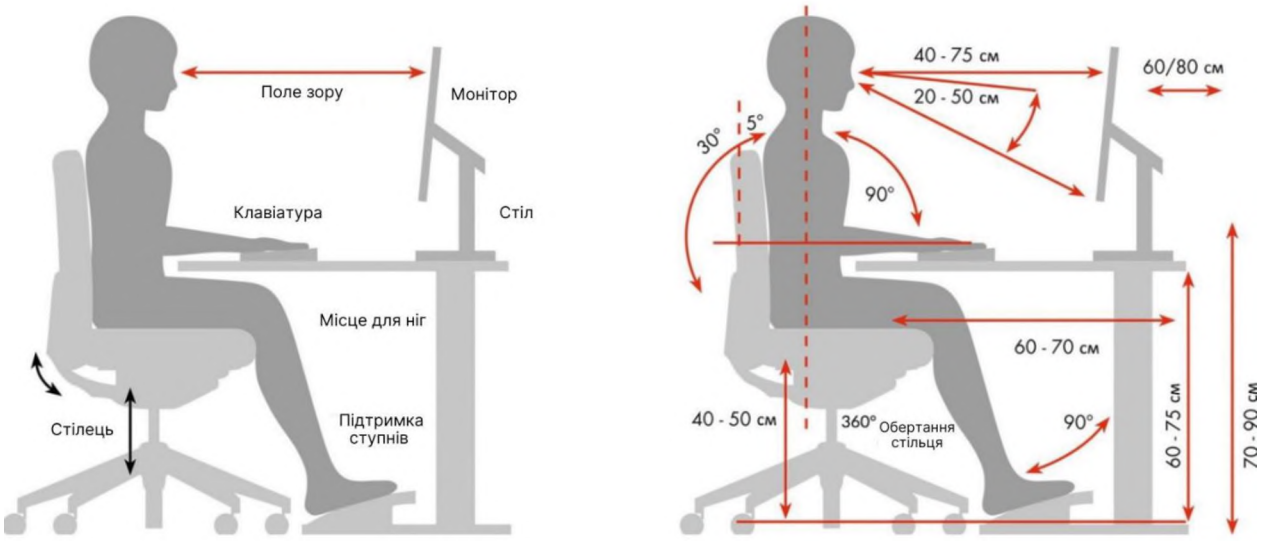


Рисунок 3.1. Правильне робоче місце

Розташування пристрою моніторингу: Дисплей пристрою діагностики (OLED-дисплей) має бути розміщений у зручному для спостереження місці, яке не вимагає зайвих рухів голови чи очей, забезпечуючи швидке візуальне зчитування даних "з першого погляду". Важливо, щоб він не було додаткових відблисків.

Ергономіка робочого столу: робочий стіл повинен мати достатні розміри для розміщення комп'ютера, пристрою моніторингу, клавіатури, миші та необхідних інструментів, забезпечуючи свободу рухів. Бажано, щоб стіл мав можливість регулювання висоти (в межах 680-800 мм), що дозволяє адаптувати його під індивідуальні особливості користувача.

Ергономіка робочого стільця: Робочий стілець має бути оснащений підйомно-поворотним пристроєм для регулювання висоти сидіння та спинки, а

також кута її нахилу. Спинка стільця повинна забезпечувати надійну опору для попереку, а сидіння — бути зручним. Наявність підлокітників, що регулюються, є бажаною для підтримки рук.

Правильна робоча поза: При роботі за комп'ютером та з електронними компонентами важливо підтримувати правильну робочу позу. Спина повинна бути прямою, опираючись на спинку крісла. Ноги повинні стояти на підлозі чи на спеціальній підставці під кутом приблизно 90%. Руки повинні бути розслаблені, а зап'ястя - прямими, розташовані на одному рівні з клавіатурою.

Розташування клавіатури та миші: Клавіатуру слід розташовувати на поверхні столу на відстані 100-300 мм від краю, зверненого до працюючого. Миша повинна знаходитись поруч з клавіатурою, забезпечуючи мінімальне напруження для руки.

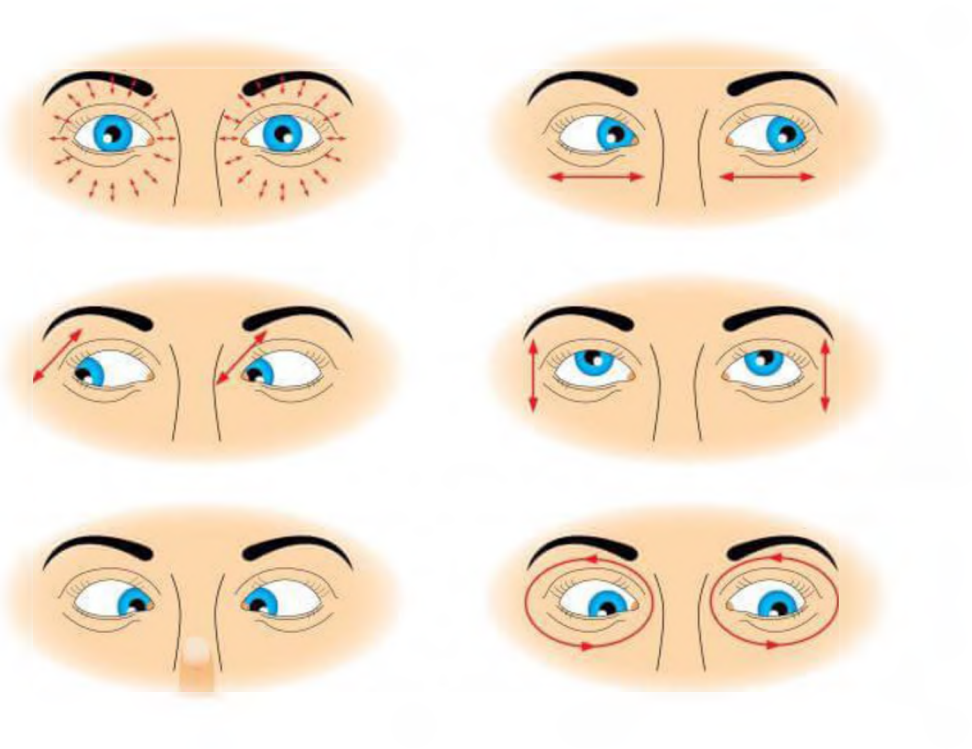


Рисунок 3.2. Гімнастика для очей

Регулярні перерви: Необхідно робити короткі, регулярні перерви (кожні 45-60 хвилин роботи — 5-10 хвилин відпочинку). Під час перерв рекомендується виконувати вправи для очей, легку розминку, зміну положення тіла для зняття напруги з м'язів.

3.2 Загальні положення з електробезпеки

Для попередження уражень електричним струмом під час роботи з компонентами пристрою, а також при використанні комп'ютерного обладнання, необхідно неухильно виконувати Правила безпечної експлуатації електроустановок споживачів, а також дотримуватися наступних вимог:

Застосовувати надійну ізоляцію: всі компоненти та з'єднувальні проводи, що можуть знаходитись під напругою, повинні мати неушкоджену та ефективну ізоляцію. Ретельно перевіряти цілісності ізоляції кабелів (зокрема USB-кабелю для Arduino) та проводів Dupont є обов'язковою перед початком роботи.

Виключати доступ до небезпечних частин: необхідно повністю виключити можливість доступу до частин устаткування, що працює під небезпечною напругою. У контексті даного проєкту, це означає використовувати справні USB-портів ПК, уникати саморобних чи пошкоджених кабелів, а також акуратне підключати компоненти Arduino без ризику коротких замикань.

Виконувати роботи лише зі знеструмленим обладнанням: всі маніпуляції, пов'язані з електричними підключеннями — приєднання/від'єднання кабелів, зміна схеми, заміна компонентів (Arduino Nano, OLED-дисплей, кабелі Dupont), — потрібно виконуватися лише при повністю знеструмленому обладнанні. Це означає, що персональний комп'ютер, до якого підключається Arduino, має бути вимкнений з електромережі, а Arduino — відключено від USB-порту.

Використовувати справний інструмент та обладнання: у роботи використовувати лише справні, не пошкоджені інструменти та обладнання. Усі інструменти, що використовуються для монтажу (наприклад, пінцети, тонкі викрутки), мають бути ізольованими, а їх рукоятки — цілими.

3.3 Пожежна безпека

Забезпечення пожежної безпеки приміщень, де є електричні мережі та електронне обладнання, регламентується Правилами пожежної безпеки в Україні. Хоча пристрій моніторингу використовує низькі напруги, ризик виникнення пожежі через короткі замикання, перегрів компонентів та несправність обладнання

завжди існує. Робоче місце, де виконуються роботи з електронною технікою, зазвичай відповідає категорії Д пожежної безпеки (приміщення, де знаходяться негорючі речовини та матеріали в холодному стані).

Як запобігти пожежі:

Контролювати технічний стан компонентів: регулярно проводити візуальний огляд усіх компонентів пристрою (Arduino Nano, OLED-дисплея, проводів) на предмет можливих пошкоджень, ознак перегріву, оплавлення чи нетипового запаху. У разі виявлення таких ознак, пристрій слід негайно відключити від живлення.

Уникати перевантажень: слідкувати за відповідністю споживаної потужності пристрою можливостям джерела живлення (USB-порту ПК). Забороняється підключати до мікроконтролера Arduino та USB-порту ПК споживачі, які перевищують максимально допустимий струм, оскільки це може призвести до перегріву та загоряння.

Дотримуватися чистоти на робочому місці: тут повинно не повинно бути сторонніх предметів, пилу та сміття, оскільки накопичення пилу на електронних компонентах може призвести до порушення тепловідведення, перегріву та, як наслідок, до коротких замикань чи загоряння.

Не зберігати горючі матеріали: заборонено зберігати легкозаймисті та горючі матеріали (папір, тканина, аерозолі, рідини) безпосередньо поблизу працюючого електронного обладнання чи у зоні можливого перегріву.

Відключати обладнання від мережі: відключати все електрообладнання від мережі, залишаючи робоче місце навіть на короткий час.

Оснащення первинними засобами пожежогасіння та пожежною сигналізацією: на робочому місці чи в безпосередній близькості повинні бути доступні первинні засоби пожежогасіння. Для гасіння електроустановок, що знаходяться під напругою (до 1000 В), слід використовувати вуглекислотні (ВВ) чи порошкові (ВП) вогнегасники. Забороняється використовувати водні та пінні вогнегасники для гасіння електрообладнання, оскільки це може призвести до

ураження електричним струмом. У приміщеннях, де розташоване робочі місця, бажано наявність установки автоматичної пожежної сигналізації.

Рисунок 3.3. Вогнегасник порошковий

Отже, розробка та використання пристрою моніторингу комп'ютера не пов'язана з підвищеним рівнем ризику за умови дотримання норм техніки безпеки, ергономіки та санітарно-гігієнічних вимог. Всі види робіт виконувалися у відповідних умовах, що дозволяє вважати проєкт безпечним для розробника та користувача.

ВИСНОВКИ

У результаті виконаної дипломної роботи було реалізовано мобільний пристрій, сумісний з актуальними представниками сімейства ОС Windows, який забезпечує отримання, обробку та візуалізацію ключових параметрів персонального комп'ютера в режимі реального часу, зокрема — навантаження на центральний процесор, графічний процесор та обсяг використаної оперативної пам'яті. Основу пристрою становить мікроконтролер Arduino Nano, який приймає інформацію, зібрану Python-скриптом з утиліти Open Hardware Monitor, через послідовне з'єднання. Дані виводяться у зручному форматі на дисплей SSD1306.

У ході роботи:

проведено технічний аналіз існуючих рішень на ринку, зокрема пристроїв з функціями системного моніторингу, і виявлено їхні недоліки з точки зору доступності, енергоспоживання та масштабованості;

обрано як апаратну платформу Arduino Nano у комбінації з OLED-дисплеєм SSD1306, що дозволило досягти балансу між функціональністю, енергоефективністю та зручністю інтеграції;

реалізовано передачу даних із застосунком Open Hardware Monitor, що стало джерелом системної інформації про роботу ПК;

розроблено окремий Python-скрипт, що забезпечує зчитування параметрів із локального сервера, їх обробку та передавання на мікроконтролер;

створено програмну частину для Arduino, яка здійснює розпізнавання вхідного сигналу та виведення показників у зручному форматі на екран пристрою.

Тестування підтвердило працездатність, стабільність і практичну користь пристрою. Він може застосовуватися як елемент персонального моніторингу ПК чи демонстраційний прилад. А також проєкт має потенціал для подальшого розвитку і масштабування, зокрема у напрямку бездротової передачі даних чи створення версії з веб-інтерфейсом.

Отже, поставлені у дипломному проєкті завдання були повністю реалізовані, а результат демонструє практичну цінність і технічну доцільність запропонованого рішення.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Баран В., Власюк Г. та ін. Основи мікропроцесорної техніки: лаб. практикум: навч. посіб. [Електронний ресурс].: https://ela.kpi.ua/server/api/core/bitstreams/d7fa0663-3988-422d-9be0-758e90c95eb0

2. Бойчик І. М., Харів П. С., Холчан М. І., Піча Ю. В. Економіка підприємства: навч. посіб. – 3-тє вид., випр. і доп. – Київ : Каравела, 2016.

3. Глухов О. В. Вивчення властивостей мікроконтролерів і електронних систем на базі платформи Ардуіно: навч. посіб.– Харків : ХНУРЕ, 2019.

4. Копей В. Б. Мова програмування Python для інженерів і науковців: навч. посіб. – Івано-Франківськ : ІФНТУНГ, 2019.

5. Кошель В., Сав'юк Г., Дзундза Б. Основи охорони праці: навч.-метод. посіб. – Івано-Франківськ : НАІР, 2020. – 182 с.

6. Крамар С. С. Arduino для вчителя інформатики: практ. посіб. – Київ : ІЦО НАПН України, 2025.

7. Обєгов М. М., Крюков І. Д. Діагностичні пристрої на базі Arduino // Інформатика, управління та штучний інтелект : тези 11-ї міжнар. наук.-техн. конф., Харків – Краматорськ – Тернопіль, 09–12 травня 2024 р. – Харків : Impress, 2024.

8. Павловський О. М., Цибульник С. О. Системи автоматизованого збору інформації: комп'ютерний практикум: навч. посіб. [Електронний ресурс]: https://ela.kpi.ua/server/api/core/bitstreams/32ba2ac8-1ce5-4ad0-a61f-4b2c25277137.

9. Роговенко А. І., Красножон О. В., Красножон А. В., Усік А. М. Архітектура комп'ютерних систем: метод. вказівки до лабораторних робіт. – Чернігів : НУ «Чернігівська політехніка», 2024.

10. Arduino Україна. Arduino Nano A000005 [Електронний ресурс]: https://arduino.ua.

11. Open Hardware Monitor. Документація [Електронний ресурс]: https://openhardwaremonitor.org/documentation.

12. Python. Офіційна документація [Електронний ресурс]: https://docs.python.org.

Додаток А. Python-скрипт для передачі даних на Arduino.

```
import requests
import serial
import time

OHM_URL = 'http://localhost:8085/data.json'
COM_PORT = 'COM3'  # ЗМІНИТИ за необхідності
BAUD_RATE = 115200

def find_value(data, target_text, parent_text=None):
    if isinstance(data, dict):
        if data.get('Text') == target_text:
            if parent_text is None or data.get('Parent') ==
parent_text:
                return  data.get('Value',  '').replace(',',
'.').replace(' %', '').strip()
        for child in data.get('Children', []):
            result    =    find_value({**child,    'Parent':
data.get('Text')}, target_text, parent_text)
            if result is not None:
                return result
    elif isinstance(data, list):
        for item in data:
            result    =    find_value(item,    target_text,
parent_text)
            if result is not None:
                return result
    return None

def main():
    print("Відкриваю послідовний порт...")
    ser = serial.Serial(COM_PORT, BAUD_RATE, timeout=1)
    time.sleep(2)

    while True:
        try:
            response = requests.get(OHM_URL)
            data = response.json()

            cpu_load = find_value(data, 'CPU Total', 'Load')
            gpu_load = find_value(data, 'GPU Core', 'Load')
            ram_load = find_value(data, 'Memory', 'Load')

            cpu_load = cpu_load or '0'
            gpu_load = gpu_load or '0'
            ram_load = ram_load or '0'
```

```
                # Тепер без процентів!
                output_str    =    f"CPU:{int(float(cpu_load))}
GPU:{int(float(gpu_load))} RAM:{int(float(ram_load))}"
                print(output_str)

                ser.write((output_str + '\n').encode())

                time.sleep(1)

            except Exception as e:
                print(f"Помилка: {e}")
                time.sleep(5)

    if __name__ == '__main__':
        main()
```

Додаток Б. Програмний код для Arduino Nano.

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_ADDRESS 0x3C

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire);

void setup() {
  Serial.begin(115200);
  delay(1000);

  if (!display.begin(SSD1306_SWITCHCAPVCC, OLED_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    while (true); // стоп, если дисплей не инициализировался
  }

  display.clearDisplay();
  display.setTextSize(2); // крупный текст
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
  display.println(F("Waiting..."));
  display.display();
}

void processData(String data) {
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
  display.println(F("Load PC:"));

  int cpuStart = data.indexOf("CPU:");
  int gpuStart = data.indexOf("GPU:");
  int ramStart = data.indexOf("RAM:");

  if (cpuStart != -1 && gpuStart != -1 && ramStart != -1) {
    String cpu = data.substring(cpuStart + 4, gpuStart);
    cpu.trim();
    String gpu = data.substring(gpuStart + 4, ramStart);
```

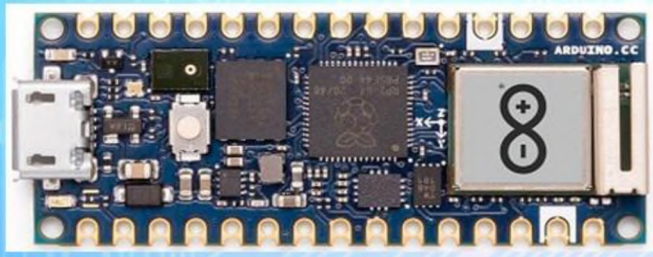
```
    gpu.trim();
    String ram = data.substring(ramStart + 4);
    ram.trim();

    display.setCursor(0, 16);
    display.print("CPU: ");
    display.print(cpu);
    display.println("%");

    display.setCursor(0, 32);
    display.print("GPU: ");
    display.print(gpu);
    display.println("%");

    display.setCursor(0, 48);
    display.print("RAM: ");
    display.print(ram);
    display.println("%");
  } else {
    display.setCursor(0, 16);
    display.setTextSize(1);
    display.println("Invalid data");
  }

  display.display();
}

void loop() {
  static String inputString = "";

  while (Serial.available()) {
    char inChar = (char)Serial.read();
    if (inChar == '\n') {
      processData(inputString);
      inputString = "";
    } else {
      inputString += inChar;
    }
  }
}
```

Додаток В. Слайди мультимедійної презентації

Розробка пристрою діагностики стану ПК на платформі Arduino

Кліновський Володимир, гр. 4КС-58

Вибір елементної бази: мікроконтролери



Arduino Nano

Основні параметри мікроконтролера Arduino Nano

Параметр	Значення
Процесор	ATmega328P
Тактова частота	16 МГц
Кількість цифрових пінів	14 (з них 6 — PWM)
UART	Є (Serial TX/RX)
Живлення	5 V (USB)
Пам'ять	32 КБ Flash, 2 КБ SRAM
Підключення по USB	через mini-USB кабель

Raspberry Pi Pico

ESP32

Arduino Uno

Вибір елементної бази: дисплеї

Основні параметри OLED-дисплея SSD1306

Параметр	Значення
Тип	OLED 0.96"
Контролер	SSD1306
Роздільна здатність	128×64 пікселів
Інтерфейс	I²C (4 контакти: VCC, GND, SDA, SCL)
Напруга живлення	3.3-5 V
Споживана потужність	~20 мА

Приклад OLED 0.96"

Приклад LCD 16x2

Приклад TFT 1.8"

Апаратна частина пристрою

Фото мікроконтролера,
що використано в проєкті

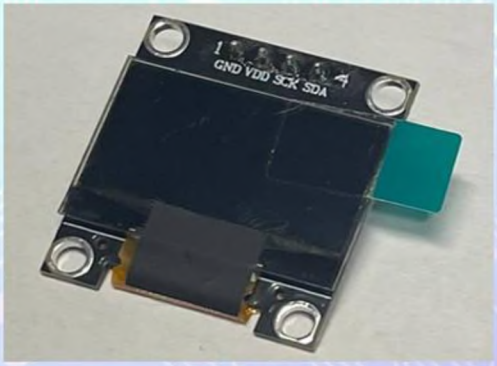


Фото дисплея,
що використано в проєкті

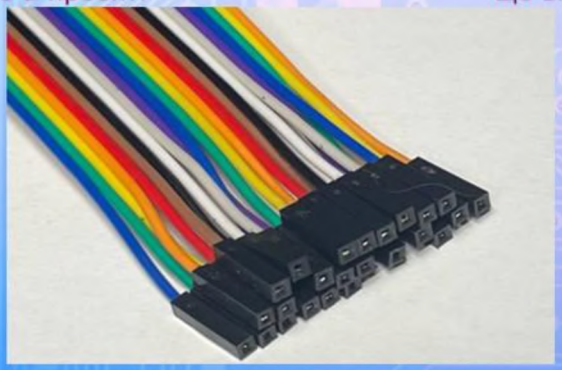
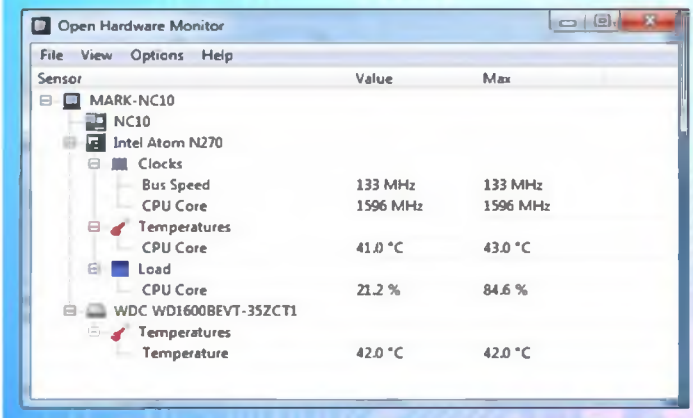


Фото кабелів Dupont, що використано в проєкті

Дослідження програмних засобів

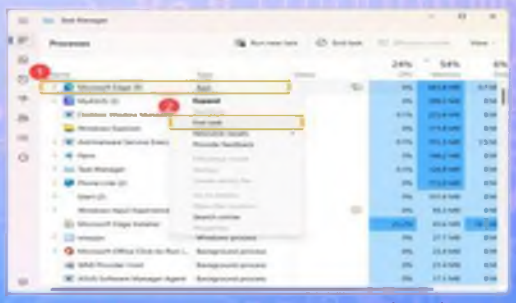


Open Hardware Monitor

Параметр	Значення
Версія	0.9.6
Сумісність	Windows XP/7/10/11
Формат експорту даних	JSON через локальний HTTP-сервер
Частота оновлення	до 1 разу на секунду
Підтримка сенсорів	CPU, GPU, RAM, HDD/SSD, материнська плата

Основні параметри використання Open Hardware Monitor

MSI Afterburner



Диспетчер завдань Windows

Загальна архітектура системи

Структурна схема пристрою діагностики

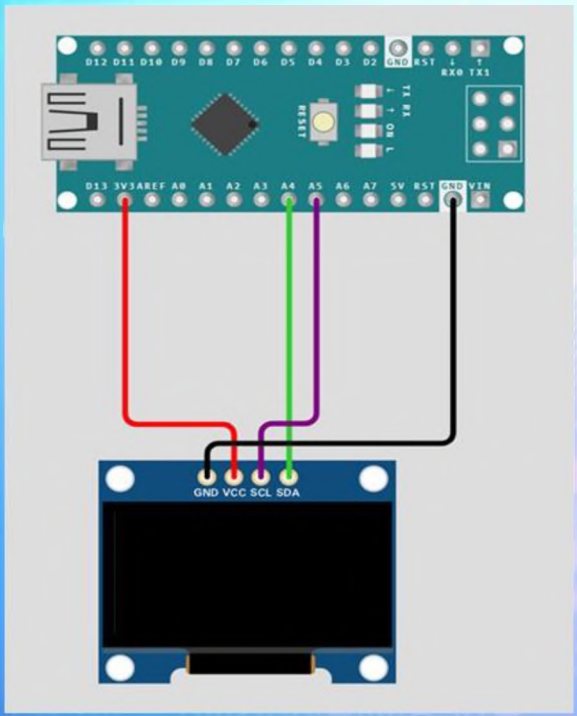
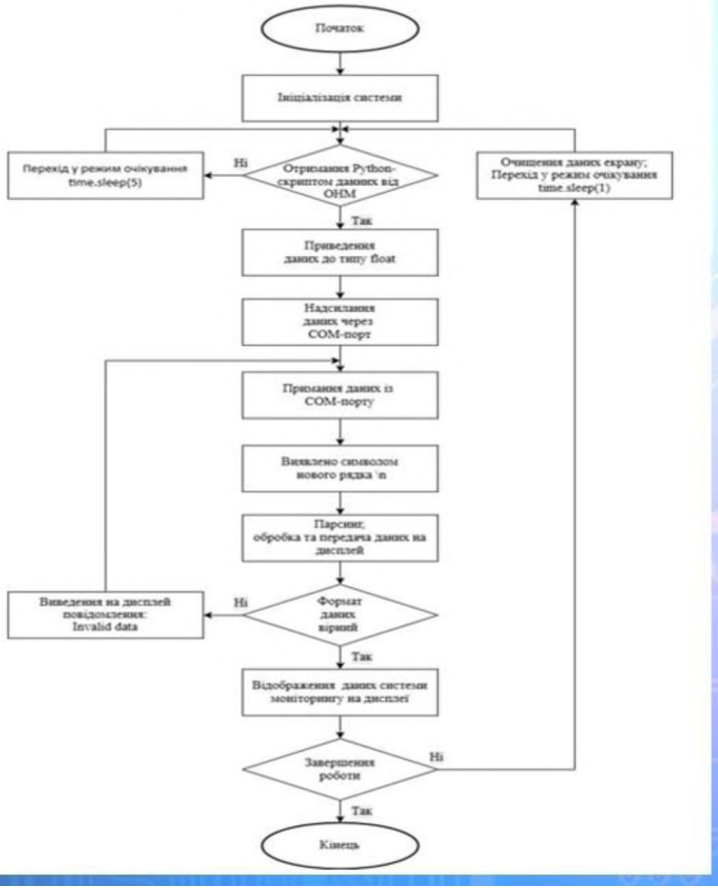


Схема підключення
OLED-дисплея до Arduino Nano

Функції контактів A4 (SDA) та A5 (SCL) в Arduino

Вивід дисплея	Вивід Arduino	Функція
SDA	A4	Передача даних (I²C)
SCK	A5	Синхронізація (I²C)
VDD	3V3	Живлення дисплея
GND	GND	Земля (спільний контур)



Узагальнена блок-схема алгоритму вимірювань

Фрагменти коду Python-скрипт для передачі даних на Arduino

Фрагмент 1:
Отримання та обробка JSON-даних з OHM у Python

```python
response = requests.get(OHM_URL)                    # Отримання JSON-даних з Open Hardware Monitor
data = response.json()                              # Десеріалізація JSON у Python-структуру

cpu_load = find_value(data, 'CPU Total', 'Load')  # Витяг значення навантаження CPU
gpu_load = find_value(data, 'GPU Core', 'Load')   # Витяг значення навантаження GPU
ram_load = find_value(data, 'Memory', 'Load')     # Витяг значення навантаження RAM
```

```
CPU:24 GPU:9 RAM:35
```

Фрагмент 2:
Приклад форматованого рядка

```python
ser = serial.Serial(COM_PORT, BAUD_RATE, timeout=1)
```

Фрагмент 3:Команда відкриття порту

Фрагменти коду для Arduino Nano

Фрагмент 1:
Підключення бібліотек і оголошення дисплея

```arduino
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_ADDRESS 0x3C

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire);
```

```arduino
void setup() {
  Serial.begin(115200);
  delay(1000);

  if (!display.begin(SSD1306_SWITCHCAPVCC, OLED_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    while (true);
  }
```

Фрагмент 2:
Налаштування дисплея у функції setup()

Фрагмент 3:
Початкове оформлення дисплея

```arduino
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(WHITE);
}
```

Фрагменти коду для Arduino Nano

```
void loop() {
  while (Serial.available()) {
    char c = Serial.read();
    if (c == '\n') processData(inputString), inputString = "";
    else inputString += c;
  }
}
```

Фрагмент 4:
Головний цикл
прийому даних

Фрагмент 5:
Обробка вхідного рядка

```
void processData(String data) {
  int cpu = data.substring(0, data.indexOf(',')).toInt();
  int gpu = data.substring(data.indexOf(',') + 1).toInt();
```

```
  display.clearDisplay();
  display.setCursor(0, 0);
  display.print("CPU: "); display.print(cpu); display.println("%");
  display.print("GPU: "); display.print(gpu); display.println("%");
  display.display();
}
```

Фрагмент 6:
Вивід даних на
дисплей

Фото зібраного пристрою після тестової компоновки

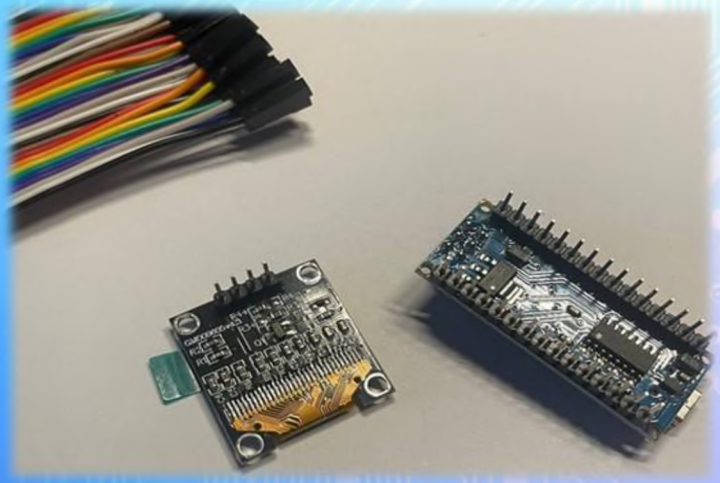
Результат
тестової збірки пристрою

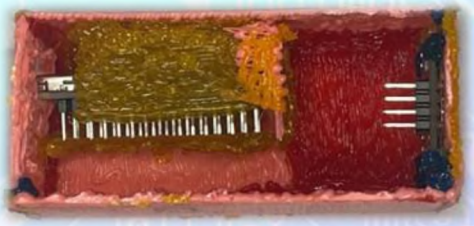
Некоректний вивід
на OLED-дисплей SSD1306

Результат коректної роботи
дисплея

Фото монтування корпусу

Компоненти проєкту до монтажу

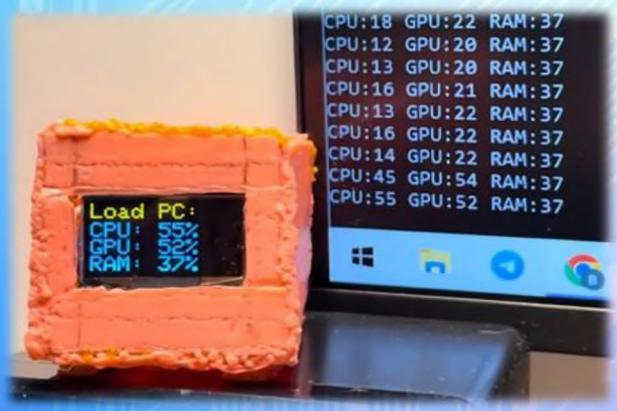




Початковий етап монтування корпусу пристрою

Завершення внутрішнього з'єднання компонентів у корпусі пристрою

Підключений до ПК пристрій



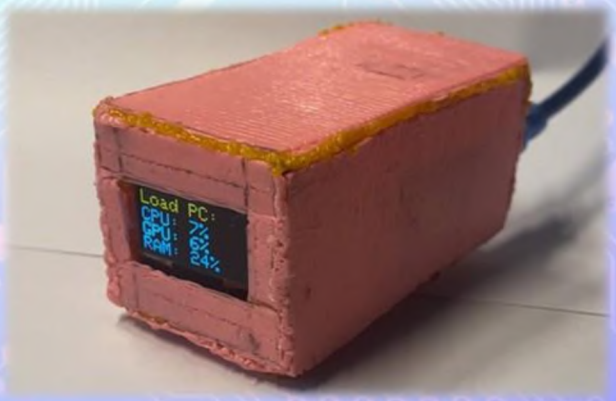
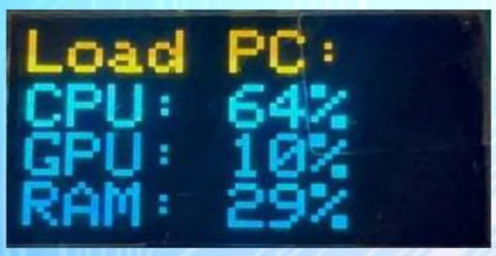


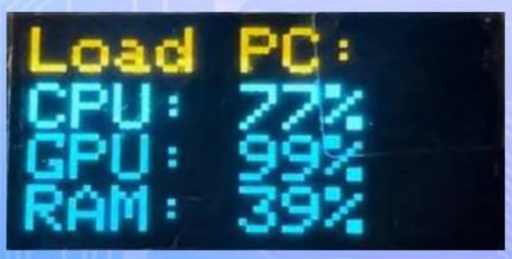
Схема тестування приладу

Імітація зміни стану ПК



Навантаження на ПК під час відкриття кількох вкладок у браузері

Оцінка роботи скрипта Python при зміні навантаження на ПК



Навантаження на ПК під час роботи гри

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Кліновського Володимира Віталійовича

(прізвище, ім`я та по батькові)

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Обслуговування комп'ютерних систем і мереж»*

Керівник дипломного проекту (роботи) *Кривченко Анастасія Анатоліївна*

(прізвище, ім`я та по батькові)

Тема дипломного проекту (роботи) *Розробка пристрою діагностики стану ПК на платформі Arduino*

Обсяг розрахунково-пояснювальної записки *92* сторінок

Обсяг графічної (презентаційної) частини *16* аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту завданню

Представлений дипломний проект відповідає затвердженій темі та виконаний відповідно технічному завданню. Дипломний проект присвячений розробці та реалізації пристрою діагностики стану ПК на платформі Arduino і складається з пояснювальної записки та мультимедійної презентації з відповідними схемами.

б) характеристика виконання кожного розділу дипломного проекту

Пояснювальна записка складається з основного розділу (аналіз застосування приладів для контролю показників системи ПК, визначення основних показників ПК, розробка моделі пристрою, підбір елементів пристрою, створення схеми та інтеграція компонентів пристрою, реалізація програмного забезпечення мікроконтролера, аналіз результатів роботи пристрою), економічного розділу, розділу охорони праці та додатків. Перелічені розділи поетапно охоплюють розробку, виконані докладно та обґрунтовано.

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту

Графічна частина складається з 16 слайдів мультимедійної презентації, виконаної у програмному продукті MS PowerPoint, які містять структурні, принципові та функціональні схеми, фото роботи пристрою діагностики стану ПК, блок-схеми алгоритмів, передбачені технічним завданням. Пояснювальна записка виконана акуратно та у відповідності до норм. Якість виконання пояснювальної записки добра, розробку виконано у повному обсязі.

г) перелік позитивних якостей дипломного проекту *У проекті реалізовано повністю відкрите ПЗ (Python + Arduino); Простота масштабування (чітко описано, як додати нові сенсори та як перенести проєкт на ESP32 для Wi-Fi версії)*

д) основні недоліки дипломного проекту *Залежність від Windows + Open Hardware Monitor: OHM офіційно не підтримує нові відеокарти/чипсети; рішення не працює «з коробки» під Linux чи macOS; time.sleep(1) у Python і delay(1000) у Arduino затримують цикл; за бажанням показувати 10+ параметрів така архітектура «задихнеться»; Присутні деякі помилки оформлення у тексті пояснювальної записки*

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Добре*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові рецензента *к.т.н. Шибаєва Наталя Олегівна*

Місце роботи і посада рецензента *Національний університет «Одеська політехніка», доцент кафедри інформаційних технологій*

Підпис: _____

«*20*» *червня* *2025 р.*

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Кліновського Володимира Віталійовича

(прізвище, ім`я та по батькові)

Спеціальність: *123 "Комп'ютерна інженерія"*

Освітньо-професійна програма: *«Обслуговування комп'ютерних систем і мереж»*

Тема дипломного проекту: *Розробка пристрою діагностики стану ПК на платформі Arduino*

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) *Дипломний проект виконано відповідно технічному завданню. Пояснювальна записка містить 92 сторінки. У пояснювальній записці наведено етапи розробки пристрою діагностики стану ПК на платформі Arduino, а також його програмного забезпечення. Графічна частина складається з 16 слайдів мультимедійної презентації, які також містять креслення, передбачені технічним завданням. Якість виконання пояснювальної записки та графічної частини добра, розробку виконано в повному обсязі.*

б) самостійність роботи над проектом: *Протягом всього строку дипломного проектування та переддипломної практики здобувач освіти Кліновський В.В. поступово та послідовно виконував всі етапи розробки. Всі роботи здобувач освіти виконував самостійно, з оглядом на рекомендації керівника*

в) теоретична підготовка випускника (випускниці): *Здобувач освіти Кліновський В.В. під час роботи над дипломним проектом вивчив достатню кількість літературних джерел та матеріалів за даною тематикою.*

Вважаю, що теоретична підготовка дипломника добра і він готовий до захисту дипломного проекту

г) вміння розв'язувати виробничі та конструкторські питання *Під час дипломного проектування здобувач освіти Кліновський В.В. мав змогу самостійно приймати окремі рішення з реалізації апаратної та програмної частини пристрою моніторингу стану ПК та показав вміння організовано працювати над поставленим завданням, розробляти структурні схеми та програмно-апаратні зв'язки із застосуванням сучасних комп'ютерних програмних засобів та САПР, таких як Arduino IDE, TinkerCAD, Python, а також готувати презентаційні та звітні матеріали.*

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові керівника дипломного проекту *Кривченко Анастасія Анатоліївна*

Місце роботи і посада керівника дипломного проекту *ВСП "Одеський технічний фаховий коледж ОНТУ", викладач спецдисциплін комісії комп'ютерних технологій та програмної інженерії, голова обласної методичної комісії викладачів компютерної інженерії*

Підпис 

«16» 06 2025 р.

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
(ДИПЛОМНОГО ПРОЕКТУ)
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Кліновський В.В.,
здобувач освіти гр. 4КС-58, та

Кривченко А.А.,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до дипломного проекту фахового молодшого бакалавра на тему:

«Розробка пристрою діагностики стану ПК на платформі Arduino» (автор роботи – Кліновський В.В., керівник роботи – Кривченко А.А.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2025 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи і даємо згоду на обробку персональних даних.

Виконавець _____ / Кліновський В.В. /

Керівник _____ / Кривченко А.А. /

«16» червня 2025 р.

ДОВІДКА

циклової комісії КТ та ПІ
про допуск до захисту дипломного проєкту
здобувача (здобувачки) освіти IV курсу
відділення комп'ютерних систем групи 4КС-58

Кліновського Володимира Віталійовича

на тему *Розробка пристрою діагностики стану ПК на платформі Arduino*

Висновок відповідальної особи за проведення нормоконтролю:

пояснювальна записка до дипломного проєкту виконана з деякими порушеннями ДСТУ, оформлена відповідно до вимог Положення про дипломне проєктування

(підпис) *16.06.2025* (дата) *Петрашова В.І.* (П.І.Б.)

Висновок відповідальної особи за перевірку роботи на наявність академічного плагіату *згідно звіту про перевірку від 13.06.2025 р. значення коефіцієнту подібності в роботі становить 7,91%, коефіцієнт цитування – 10,15%.*

(підпис) *16.06.2025* (дата) *Краснокутська К.Г.* (П.І.Б.)

Попередня експертиза (малий захист) дипломного проєкту

здобувача (здобувачки) освіти *Кліновського В.В.* (П.І.Б.)

проведена «*16*» *червня* 20*25* р.

Висновки *Пояснювальна записка до дипломного проєкту виконана у повному обсязі. Випускна кваліфікаційна робота (дипломний проєкт) відповідає вимогам Положення про дипломне проєктування та рекомендована до захисту.*

Голова ЦК КТ та ПІ (підпис) *Кривченко Ю.В.* (П.І.Б.)

StrikePlagiarism.com

Дата звіту 6/13/2025
Дата редагування 6/13/2025

Документ прийнятий

Звіт подібності

метадані

Назва організації
Odesa Technical Professional College of Odesa National University of Technology
Заголовок
Розробка пристрою діагностики стану ПК на платформі Arduino

Автор
Кліновський Володимир Віталійович
Науковий керівник / Експерт
Кривченко Анастасія Анатоліївна

підрозділ
Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету"

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

КП 1

КЦ

25	17266	118690
Довжина фрази для коефіцієнта подібності 2	Кількість слів	Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв	R	5
Інтервали	A→	0
Мікропробіли	·	0
Білі знаки	R	0
Парафрази (SmartMarks)	a	84

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content	99 0.57 %
2	https://studopedia.org/7-163419.html	56 0.32 %
3	https://card-file.ontu.edu.ua/bitstreams/1dff552d-7200-49b8-ae1d-ba76a1335685/download	36 0.21 %
4	https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content	33 0.19 %
5	https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content	30 0.17 %

6	https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content	28 0.16 %
7	https://card-file.ontu.edu.ua/bitstreams/29489599-0581-4ce6-8890-c3b13d9f2e0e/download	27 0.16 %
8	https://e-journals.npu.edu.ua/index.php/ikt/article/download/111/pdf/	27 0.16 %
9	https://card-file.ontu.edu.ua/bitstreams/2b75599e-e1ac-412d-bf09-10d2eb49022f/download	24 0.14 %
10	https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content	24 0.14 %

з домашньої бази даних (0.14 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Розробка 3D-гри у жанрі survival-horror з налаштуваннями рівнів складності [redacted] 6/12/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	25 (3) 0.14 %

з програми обміну базами даних (0.25 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	РОЗРОБКА СИСТЕМИ БІОМЕТРИЧНОЇ ІДЕНТИФІКАЦІЇ КОРИСТУВАЧА ЗА РЕЗУЛЬТАТАМИ АНАЛІЗУ СТИЛЮ НАБОРУ ТЕКСТУ НА КЛАВІАТУРІ КОМП'ЮТЕРА [redacted] 6/11/2024 Odessa National Polytechnic University (ІІБРТ, Каф. кібербезпеки та програмного забезпечення)	15 (1) 0.09 %
2	Організація перевезення готової продукції від ПРАТ «ПТАХОКОМБІНАТ «БЕРШАДСЬКИЙ»» с. Війтівка до торговельної мережі Вінницької, Черкаської і Одеської областей транспортом ТОВ «ЖОРНЯКИ» м. Бершадь Вінницької області [redacted] 12/27/2023 Haivoron Polytechnic Applied College (Haivoron Polytechnic Applied College)	9 (1) 0.05 %
3	2018_6030507_Moskviak_Ilona_Ruslanivna_51995 [redacted] 10/26/2024 National University "Lviv Politechnika" (National University Lviv Politechnika)	5 (1) 0.03 %
4	Добровольська Ольга_диплом23.12.doc [redacted] 12/26/2022 Odessa National Polytechnic University (ІМБ, Кафедра ДММОМ)	5 (1) 0.03 %
5	Zastosowanie Arduino do laserowych i ultradźwiękowych pomiarów odległości [redacted] 1/28/2022 Politechnika Lubelska (Wydział Elektrotechniki i Informatyki)	5 (1) 0.03 %
6	CNUT/ Особливості_застосування_припущення_про_безперервність_при_аудиті_підприємств_космічної_галузі____Cy.pdf [redacted] 8/28/2017 National University Chernihiv Politechnika (NUCP) course papers (Deanery)	5 (1) 0.03 %

з Інтернету (7.51 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	**https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content**	**334 (17) 1.93 %**
2	https://card-file.ontu.edu.ua/bitstreams/aed610a6-43ef-47e0-9066-e85c89456f3e/download	85 (10) 0.49 %

3	**https://card-file.ontu.edu.ua/bitstreams/1dff552d-7200-49b8-ae1d-ba76a1335685/download**	**74 (5) 0.43 %**
4	https://d.lib.msu.edu/etd/46913/datastream/FULL_TEXT/view/	64 (10) 0.37 %
5	**https://card-file.ontu.edu.ua/bitstreams/29489599-0581-4ce6-8890-c3b13d9f2e0e/download**	**61 (4) 0.35 %**
6	https://kneu.edu.ua/userfiles/d-26.006.02/2015/Chalyuk_dis.pdf	56 (6) 0.32 %
7	**https://studopedia.org/7-163419.html**	**56 (1) 0.32 %**
8	https://www.rivneprod.gov.ua/wp-content/uploads/2020/06/Dodatok-do-Rozporyadzhennya-Golovnogo-derzhavnogo-veterynarnogo-inspektora-Ukrayiny-vid-25-travnya-2020-r.-41.pdf	53 (7) 0.31 %
9	**https://e-journals.npu.edu.ua/index.php/ikt/article/download/111/pdf/**	**46 (3) 0.27 %**
10	https://d.lib.msu.edu/etd/46945/datastream/FULL_TEXT/view/	44 (7) 0.25 %
11	https://card-file.ontu.edu.ua/bitstreams/bbaf3f38-16a8-4070-bead-5562769b7c71/download	42 (4) 0.24 %
12	https://card-file.ontu.edu.ua/bitstreams/c1f3e592-1123-419d-b14a-4c28662f0f1e/download	36 (4) 0.21 %
13	https://card-file.ontu.edu.ua/server/api/core/bitstreams/fc8a1853-39fc-4671-8807-2fd27ddb0779/content	32 (3) 0.19 %
14	https://card-file.ontu.edu.ua/bitstreams/2b75599e-e1ac-412d-bf09-10d2eb49022f/download	30 (2) 0.17 %
15	https://card-file.ontu.edu.ua/server/api/core/bitstreams/995bdcec-4e4d-4321-8070-4d6badcb8e49/content	26 (2) 0.15 %
16	https://card-file.ontu.edu.ua/bitstreams/549ee9fe-7574-4ae5-b500-9fe2711f33e6/download	24 (2) 0.14 %
17	https://card-file.ontu.edu.ua/bitstreams/62baa43e-b968-4993-bb54-8cf8761a89b2/download	24 (2) 0.14 %
18	http://www.anamnesis.info/sites/default/files/Spasov_R_statiya_Anamneza_kn_1_2017%20%281%29.pdf	23 (4) 0.13 %
19	https://card-file.ontu.edu.ua/server/api/core/bitstreams/a05c07c5-bf65-4cb0-bdfa-e28694707551/content	17 (2) 0.10 %
20	https://card-file.ontu.edu.ua/bitstreams/b1c4b329-c3e8-4b5a-a1fc-ae232ec677bd/download	17 (2) 0.10 %
21	https://elartu.tntu.edu.ua/bitstream/lib/42517/1/dyplom_Perch%20V_2023.pdf	16 (2) 0.09 %
22	https://ua-referat.com/uploaded/ministerstvo-osviti-i-v3/index4.html	15 (1) 0.09 %
23	https://semcneil.github.io/Fundamentals-of-Microcontrollers-Manual/Fundamentals-of-Microcontrollers.pdf	14 (2) 0.08 %
24	https://card-file.ontu.edu.ua/bitstreams/341a820e-d025-42f3-b7dc-27e831d6c66f/download	13 (1) 0.08 %
25	http://dspace.oneu.edu.ua/jspui/bitstream/123456789/5061/1/%D0%95%D0%BA%D0%BE%D0%BD%D0%BE%D0%BC%D1%96%D1%87%D0%BD%D0%B5%20%D0%BE%D0%B1%D2%91%D1%80%D1%83%D0%BD%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%20%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D1%83%20%D1%81%D1%82%D0%B2%D0%BE%D1%80%D0%B5%D0%BD%D0%BD%D1%8F%20%D0%BA%D0%B0%D1%84%D0%B5.pdf	12 (1) 0.07 %
26	https://card-file.ontu.edu.ua/bitstreams/7b1e10b9-0ac2-4b07-afc4-8cdfff7db780/download	12 (1) 0.07 %
27	https://card-file.ontu.edu.ua/bitstreams/21173711-5b67-4b87-b17f-6302c25e7a31/download	11 (1) 0.06 %
28	https://repository.lnup.edu.ua/jspui/bitstream/123456789/676/1/Pavliukovych_bach.pdf	11 (1) 0.06 %
29	https://stryi-rada.ukraina.org.ua/sites/stryi-rada.ukraina.org.ua/files/inline-files/kolektivniy-dogovir-.docx	10 (1) 0.06 %
30	https://card-file.ontu.edu.ua/bitstreams/e8cf355e-894f-43e6-9a6b-7b3b00346401/download	10 (1) 0.06 %
31	https://simple-circuit.com/blue-pill-stm32-ssd1306-oled-i2c-mode-interface/	9 (1) 0.05 %
32	http://www.managerhelp.org/hoks-703-1.html	8 (1) 0.05 %
33	https://card-file.ontu.edu.ua/bitstreams/82a6d375-2b69-4233-b80f-fbfd149b7747/download	6 (1) 0.03 %

34 https://lastminuteengineers.com/oled-display-arduino-tutorial/ 5 (1) 0.03 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»
Спеціальність: 123 «Комп'ютерна інженерія»
Освітньо-професійна програма: «Обслуговування комп'ютерних систем і мереж» Група: 4КС- 58
Дипломний **проект здобувача освіти денної форми навчання КС.** 58.12.000.ДП
КЛІНОВСЬКОГО
ВОЛОДИМИРА ВІТАЛІЙОВИЧА
м. Одеса
2025 р.

Спеціальність: 123 «Комп'ютерна інженерія»
Освітньо-професійна програма: «Обслуговування **комп'ютерних систем і мереж»**
Група: 4КС-58
ПОЯСНЮВАЛЬНА ЗАПИСКА
до дипломного проекту на тему:
Розробка пристрою діагностики стану ПК
на платформі A^uino
Проектний матеріал складається з пояснювальної записки на 92 сторінках та
графічного (презентаційного) матеріалу на 16 аркушах (слайдах)
Дипломник _____ (Кліновський В.В.)
Керівник _____ (Кривченко А.А.)
Консультанти:
з економічного розділу_____
з розділу охорони праці та техніки безпеки
з нормоконтролю_____
старший консультант_____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»
До захисту допущений
Голова циклової комісії (Кривченко Ю.В.)
Завідувач відділення (Краснокутська К. **Г.)**
Захист « » 2025 **р. Протокол ЕК No Оцінка ЕК Секретар** ЕК
(Канський М. Ю.)
(Чорновол Н.І.)
(Петрашова В.І.)
(Кривченко Ю.В.)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»
Відділення_____ комп 'ютерних систем_____ Комісія_____ КТ та ПІ
С.петтіяльність **123 «Комп 'ютерна інженерія»** Освітньо -про фесійна програма **« «Обслуговування комп** юктерних систем і мережо»
Зяст. дир. з НВР
Беркань І.В.
ЗАТВЕРДЖУЮ:
2025 р.
здобувячеві (здобувячці) освіти
ЗАВДА ННЯ
на дипломнии проект
Кліновському Володимиру Віталійовичу
(прізвище, ім'я, по батькові)
1. Тема проекту Розробка пристрою діагностики стану ПК на платформі Агсііиію
затверджена наказом по коледжу від "____ "_____202___ р.
No_____
2. Термін здачі закінченого проекту
3. Вихідні данні до проекту 1. У якості мікроконтролеру для обчислювальної платформи використовувати Arduino Nano. 2. Візуалізувати ключові параметри ПК у режимі реального