НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут атомної та теплової енергетики
Кафедра інженерії програмного забезпечення в енергетиці

«На правах рукопису»
УДК_004.4

ДО ЗАХИСТУ ДОПУЩЕНО
В.О. Завідувача кафедри
Олександр КОВАЛЬ
“___”_____2024 р

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою
«Інженерія програмного забезпечення
інтелектуальних кібер-фізичних систем в енергетиці»
зі спеціальності 121 Інженерія програмного забезпечення
на тему: «Реєстратор вібрації поверхні на базі апаратно-програмних засобів Arduino».

Виконав: студент 2 курсу, групи ТВ-22мп
Редько Денис Вадимович _____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник доцент, д.т.н, професор Федорова Н.В. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ — 2024

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО"

Навчально-науковий інститут атомної та теплової енергетики
Кафедра інженерії програмного забезпечення в енергетиці
Рівень вищої освіти другий, магістерський
Спеціальності 121 Інженерія програмного забезпечення
Освітньо-професійна програма «Інженерія програмного забезпечення інтелектуальних кібер - фізичних систем в енергетиці»

ЗАТВЕРДЖУЮ
В.о. Завідувача кафедри
_____ Олександр КОВАЛЬ
(підпис)
«____» _____ 2024р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

Редьку Денису Вадимовичу
(прізвище, ім'я, по батькові)

1. Тема дисертації «Реєстратор вібрації поверхні на базі апаратно-програмних засобів Arduino»
Науковий керівник Федорова Н.В. д.т.н., доцент, професор кафедри ІПЗЕ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від "06" листопада 2023 року №5152-с
2. Строк подання студентом дисертації 10 січня 2024
3. Об'єкт дослідження алгоритм порівняння даних з сенсорів коливань
4. Предмет дослідження програмний і апаратний продукт для моніторингу за структурними змінами у металевих конструкціях
5. Перелік питань, які потрібно розробити проаналізувати методи моніторингу за структурними змінами у металевих конструкціях. Розробити алгоритм порівняння аналогових даних з сенсорів. Розробити апаратне рішення для збору інформації про стан об'єкту. Спроектувати архітектуру програмних компонентів. Розробити програмне забезпечення для візуалізації та контролю за збором даних.
6. Орієнтований перелік ілюстративного матеріалу: зображення подібних систем, схема функціонування програми, організаційні схеми модулів системи, діаграми класів для системних модулів та візуалізація роботи системи.
7. Орієнтовний перелік публікацій: Федорова Н.В., Редько Д.В. , Поверхневий вібраційний реєстратор на основі апаратного тапрограмного забезпечення Arduino.

Modern engineering and innovative technologies — 2022. — 78с — Електронне видання.

8. Дата видачі завдання «01» листопада 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строки виконання етапів магістерської дисертації	Примітка
1	Отримання завдання	01.11.2022	виконано
2	Дослідження предметної області	01.11.2022 - 01.12.2022	виконано
3	Постановка вимог до проєктування системи	01.12.2022 - 15.12.2022	виконано
4	Дослідження існуючих рішень	15.12.2022 - 03.01.2023	виконано
5	Підготовка публікацій	03.01.2023 - 03.03.2023	виконано
6	Розробка програмного продукту	03.03.2023 - 20.09.2023	виконано
7	Тестування	20.09.2023 - 15.10.2023	виконано
8	Захист програмного продукту	16.10-2023 - 20.10.2023	виконано
9	Підготовка магістерської дисертації	21.10.2023 - 17.11.2023	виконано
10	Передзахист	18.12.2023 - 22.12.2023	виконано
11	Захист	15.01.2024 - 19.01.2024	виконано

Студент _____ Редько Д.В.
(підпис) (прізвище та ініціали)

Науковий керівник _____ Федорова Н.В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Структура та обсяг дисертації. Магістерська дисертація складається зі вступу, п`яти розділів, висновку, переліку посилань з 24 найменувань, містить 36 рисунків, 10 таблиць. Повний обсяг магістерської дисертації складає 99 сторінки, з яких перелік посилань займає 2 сторінки, додатки займають 25 сторінок.

Актуальність. Ріст нашого сучасного суспільства неможливий без ефективної інфраструктури, яка включає в себе мости, будівлі, інженерні споруди та інші металеві конструкції. Вони створюють основу для безпеки та комфорту мільйонів людей щодня. Проте ці конструкції піддаються впливу різних факторів, таких як вібрації від транспорту та природних явищ, навантаження і знос від корозії. Існує постійне завдання контролювати стан конструкцій із металу, аби попереджати можливі пошкодження та деформації.

Вибухи, землетруси, зовнішні впливи та навіть невеликі вібрації можуть призвести до руйнування або деформації металевих споруд. Це може стати причиною надзвичайних ситуацій, загрожуючи не лише життю та безпеці людей, але й суттєвими економічними втратами. Пошкодження, завдані такими подіями, можуть бути складними у ремонті, тому важливо розробити систему моніторингу, яка б забезпечувала раннє виявлення небезпечного стану споруди та вчасну реакцію на будь-які аномалії.

Метою роботи є створення програмного та апаратного продукту для моніторингу за станом об’єктів інфраструктури зроблених з металу, що дозволить виконувати оперативний контроль за станом конструкцій, допоможе вчасно виявляти потенційні проблеми та мінімізувати ризик пошкоджень чи втрат через несправності у металевих структурах.

Об’єктом дослідження є алгоритм порівняння даних з сенсорів коливань розміщених на металевій конструкції.

Предметом дослідження є програмний і апаратний продукт для моніторингу за структурними змінами у металевих конструкціях.

Методи дослідження. В роботі використовуються сучасні технології та методи аналізу великої кількості даних, а також системи для розроблення прототипів апаратної частини. Насамперед, мови програмування C#, C фреймворки .NET, Arduino, бібліотека GnuPlot, а також файлові сховища інформаціїї.

Публікації. Федорова Н.В., Редько Д. В. Поверхневий вібраційний реєстратор на основі апаратного та програмного забезпечення Arduino /
Прийнято до друку у «Moderntechno» випуск 21 частина 1.

Практичне значення одержаних результатів полягає в отриманні програмного забезпечення та пристрою, який відносно легко можна встановити на об’єкт інфраструктури, і почати відстежувати його стан, а також отримувати сповіщення, якщо цей стан змінюється.

Ключові слова: Arduino, реєстрації вібрацій, інтелектуальний аналіз даних, сенсори на п'єзоелементах, генератори вібрацій.

ABSTRACT

The structure and scope of the dissertation. The master's thesis consists of an introduction, five chapters, a conclusion, a list of references from 24 titles, contains 36 figures, 10 tables. The full volume of the master's thesis is 99 pages, of which the list of references occupies 2 pages, the appendices occupy 25 pages.

Significance of the topic. The growth of our modern society is impossible without effective infrastructure, which includes bridges, buildings, engineering structures and other metal structures. They provide the foundation for the safety and comfort of millions of people every day. However, these structures are exposed to various factors, such as vibrations from traffic and natural phenomena, loads and wear from corrosion. There is a constant task of monitoring the condition of metal structures in order to prevent possible damage and deformation.

Explosions, earthquakes, external influences and even small vibrations can lead to the destruction or deformation of metal structures. This can lead to emergency situations, threatening not only people's lives and safety, but also significant economic losses. Damage caused by such events can be difficult to repair, so it is important to develop a monitoring system that would ensure early detection of the dangerous condition of the structure and timely response to any anomalies.

The purpose of the work is to create a software and hardware product for monitoring the state of infrastructure objects made of metal, which will allow for operational control of the state of structures, help to detect potential problems in time and minimize the risk of damage or losses due to malfunctions in metal structures.

The object of the study is an algorithm for comparing data from vibration sensors placed on a metal structure.

The subject of the study is a software and hardware product for monitoring structural changes in metal structures.

Research methods. The work uses modern technologies and methods of analyzing a large amount of data, as well as systems for developing hardware

prototypes. First of all, programming languages C#, C frameworks .NET, Arduino, GnuPlot library, as well as file storage of information.

Publications. Fedorova N.V., Redko D.V. Surface vibration recorder based on Arduino hardware and software / Accepted for publication in «Moderntechno» issue 21 part 1.

The practical value of the results is to get software and a device that can be installed relatively easily on an infrastructure object and start monitoring its state and get notified if that state changes.

Keywords: Arduino, vibration registration, intelligent data analysis, piezoelectric sensors, vibration generators.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення.

IDE – integrated development environment, інтегроване середовище розробки

Фреймворк – комплекс програмних рішень, що надає функціонал для розробки складних систем.

JIT-компілятор – технологія компіляції, коли байт-код перетворюється в машинний код під час виконання програми.

.NET – це платформа від Microsoft, яка дозволяє створювати програмні додатки.

Breadboard – це універсальна друкована плата для складання і моделювання прототипів електронних пристроїв.

СУБД — набір взаємопов'язаних даних і програм для доступу до цих даних. Надає можливості створення, збереження, оновлення та пошуку інформації в базах даних з контролем доступу до даних.

SWOT аналіз – це ефективний інструмент бізнес-планування, який використовується в бізнесі для формування стратегій. Цей інструмент допомагає проаналізувати внутрішні фактори (сильніта слабкі сторони), які впливають, і зовнішні фактори (можливості та загрози), які можуть мати вплив на організацію.

ВСТУП

Металеві структури є ключовими складовими в сучасних інженерних рішеннях та будівництві інфраструктури. Навантаження, знос та постійні вібрації можуть призвести до деформацій, пошкоджень чи мікротріщин які у свою чергу порушують цілісність конструкцій та у разі накопичення загрожують безпеці. Ця проблема потребує надійного та ефективного методу контролю, який дозволить виявляти небезпечні пошкодження та усувати їх завчасно. Нагляд за їхнім станом – основна умова для забезпечення безпеки та надійності використання, оскільки інколи навіть найдрібніші пошкодження можуть спричинити серйозні аварії та катастрофи.

Щоб вирішити цю проблему, був розроблений пристрій на основі апаратної частини Arduino, а також програма для аналізу даних з сенсорів, які у поєднанні виконують постійну задачу відстеження та аналізу змін у структурі конструкцій. Цей пристрій, заснований на апаратних та програмних рішеннях, реєструє та аналізує коливання в конструкціях. Його апаратна частина, що базується на платформі Arduino, використовує п'єзоелементи для реєстрації коливань і електромотори для генерації вібрацій. Розроблений пристрій є системою, що включає значну кількість п'єзоелектричних сенсорів, розташованих на всій поверхні металевої конструкції, а також деякої кількості, яка залежить від задачі, малих генераторів вібрацій. Цей пристрій функціонує у режимі циклічного моніторингу, який складається з кількох етапів, що забезпечують виявлення та аналіз можливих дефектів у металевих конструкціях.

Перший етап циклу активує генератори вібрацій, які породжують коливання, що розповсюджуються по всій конструкції. Під час цього процесу п'єзоелектричні сенсори фіксують мінімальні зміни у вібраціях, реагуючи на різноманітні динамічні параметри графіку коливань.

Після збору інформації генератори вібрацій припиняють функціонування, а система переходить до етапу аналізу. Отримані дані піддаються ретельному

аналізу методом порівняння з результатами минулих тестів. Це дозволяє виявити можливі аномалії, невідповідності та зміни у структурі конструкції.

Цей пристрій надає можливість в реальному часі відслідковувати стан металевих конструкцій, збираючи дані про коливання та інші параметри. Його програмна частина проводить аналіз цих даних, порівнює їх зі стандартними значеннями і сповіщає оператора про будь-які аномалії чи зміни у стані.

Такий підхід не лише дозволяє відслідковувати стан металевих конструкцій, але й своєчасно реагувати на будь-які відхилення від норми, що допомагає зменшити ризик аварій.

1 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Постановка задачі

Метою роботи є реалізація програмного та апаратного рішення, що забезпечує моніторинг за станом металевих конструкцій, що у свою чергу призводить до покращення безпеки, надійності та ефективності експлуатації інфраструктурних об'єктів.

Основні вимоги до системи будуть такі:

1) чутливість і точність: система повинна мати високу чутливість до навіть найменших змін у структурі металу, таких як мікротріщини чи деформації. Точність вимірів також є критичною, оскільки система повинна точно визначати розмір, форму та характеристики потенційних дефектів;
2) надійність та стабільність: система моніторингу металевих конструкцій має функціонувати безперебійно та стабільно, навіть у варіативних умовах навколишнього середовища, забезпечуючи постійний та надійний моніторинг;
3) швидкодія: швидкість збору даних та аналізу має бути високою для оперативного виявлення та реакції на будь-які потенційні дефекти чи зміни у металевій конструкції, що мінімізує ризик аварій;
4) автоматизований аналіз: методи аналізу та інтерпретації даних повинні бути автоматизованими для спрощення процесу виявлення дефектів та забезпечення консистентності результатів;
5) система звітності: можливість генерувати звіти та зберігати дані для подальшого вивчення є важливою для контролю за станом конструкцій у часі, а також для архівування даних для подальшого аналізу;
6) модульність та масштабованість: система моніторингу повинна бути гнучкою, щоб вона могла бути адаптована до різних розмірів

конструкцій та об'єктів, а також враховувати різні потреби в моніторингу;

7) безпека та конфіденційність: забезпечення захисту даних та конфіденційності є невід'ємною частиною, особливо коли мова йде про моніторинг у критичних секторах, що вимагають високого рівня захисту від несанкціонованого доступу та зберігання конфіденційної інформації.

1.2 Огляд існуючих рішень

Огляд існуючих рішень є ключовим етапом перед створенням будь-якого програмного забезпечення. Цей підхід суттєво зменшує ризик розробки непотрібного програмного продукту, сприяючи використанню аналогів та мінімізуючи втрату часу та ресурсів. Крім того, огляд існуючих продуктів має вагомий вплив на майбутні можливості функціоналу, оскільки в процесі огляду враховуються ідеї, які можуть бути цікавими для подальшої інтеграції та розширення запланованих функцій.

На даний момент часу в світі існує доволі багато рішень таких систем. На жаль моніторинг інфраструктури – це, зазвичай, закриті комерційні проєкти, і інформації у відкритому доступі про них мало, оскільки їх цільова аудиторія це або великі будівельні компанії або уряди країн. У сегменті доступному для придбання пересічному громадянину існують системи для виявлення точкових дефектів, вони називаються дефектоскопами, деякі з них працюють за схожим принципом, але обмежені у масштабі.

Дефектоскоп – це пристрій для неруйнівного контролю, призначений для виявлення дефектів у матеріалах, виробах або конструкціях без необхідності їх руйнівного розгляду. Його основна функція полягає в ідентифікації внутрішніх або поверхневих несправностей, таких як тріщини, пухлини, включення чи інші

аномалії, які можуть виникнути під час виробництва, експлуатації чи транспортування матеріалів та виробів.

Існують різні види дефектоскопів, зокрема:

1) вихрострумові дефектоскопи: вони використовують електромагнітні поля для виявлення дефектів у провідних матеріалах. Зміна вихрострумів відображає відмінності у властивостях матеріалу, що може свідчити про наявність дефектів;
2) магнітні дефектоскопи: ці пристрої використовують магнітні поля для виявлення дефектів у феромагнітних матеріалах, таких як зварні з'єднання чи металеві конструкції. Вони спроможні виявляти тріщини, недоліки зварювання та інші несправності;
3) ультразвукові дефектоскопи: вони використовують ультразвукові хвилі для проникнення в матеріал та виявлення дефектів. Зміни відбиваються від недоліків, інформація про які потім аналізується для визначення характеристик дефектів;
4) рентгенівські дефектоскопи: вони використовують рентгенівське випромінювання для проникнення в матеріал та створення зображень внутрішньої структури, що дозволяє виявляти дефекти.

Ці пристрої мають важливе значення для забезпечення якості та безпеки матеріалів та виробів у різних галузях, від металургії до авіаційної та автомобільної промисловості.

Порівняємо інформацію про ринок дефектоскопів, оскільки мій пристрій можна віднести до них, інформація взята з сайту cognitivemarketresearch.com [1,2,3], який частково надає доступ до звіту «Global Ultrasonic Flaw Detector Market Report 2023 Market Size Split by Type».

Серед лідерів ринку у звіті виділяють такі компанії:

- Measurement Control (США);
- Olympus (Японія);
- Sonatest (Велика Британія);

- Sonotron NDT (Ізраїль);
- Karldeutsch (Германія);
- Proceq (Швейцарія);
- Zetec (США);
- Centurion NDT (США);
- Nova Instruments (США);
- Hitachi Power Solutions (Японія);
- Modsonic (Індія);
- RYOSHO (Японія);
- KJTD (Японія);
- Novotest (Україна).

На рисунку 1.1 зображено розподіл ультразвукових дефектометрів за використанням, за 2018, 2023 та прогноз на 2030 рік.

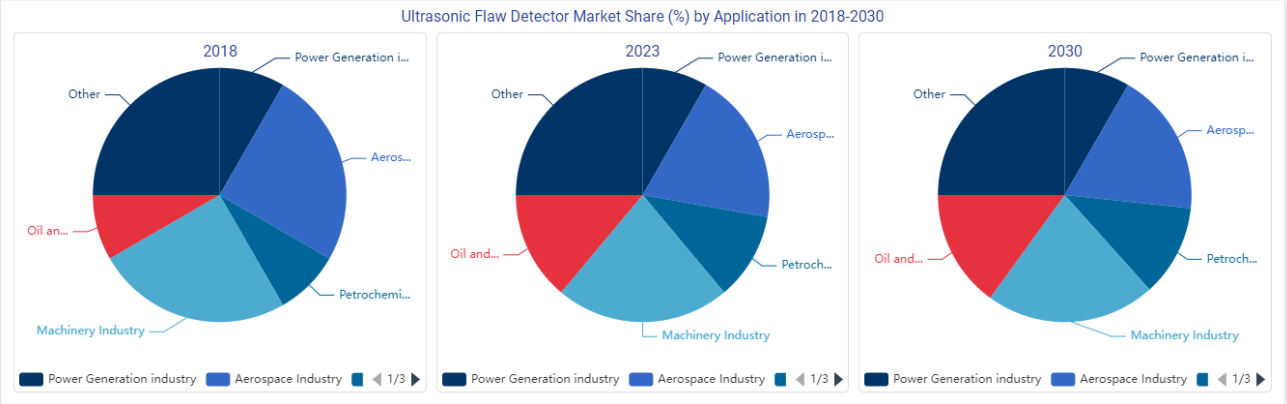


Рисунок 1.1 – Діаграма розподілу ринку за використанням

Найпопулярніші сфери використання схожих пристроїв, це:

1) енергетика;
2) аерокосмічна промисловість;
3) нафтохімічна промисловість;
4) машинобудування;
5) нафта і газ;

Як можна помітити з розподілу використання для будівельної сфери не є розповсюдженим і підпадає під загальну категорії. Це викликано тим що більшість існуючих компаній не виготовляє дешевих рішень для моніторингу за металевими конструкціями, а більш зосереджені на більш глибокому досліджень окремих ділянок конструкцій. Отже ніше для представленого пристрою виглядає відносно вільною на дуже конкурентному ринку дефектометрів.

1.2.1 GE Measurement Control

Компанія GE Measurement & Control[4] (зокрема, раніше відома як GE Inspection Technologies) спеціалізується на розробці та виробництві пристроїв для неруйнівного контролю та вимірювань у різних галузях промисловості. Вона є частиною корпорації General Electric (GE), яка займає провідні позиції у багатьох галузях, включаючи енергетику, охорону здоров'я, транспорт і багато інших.

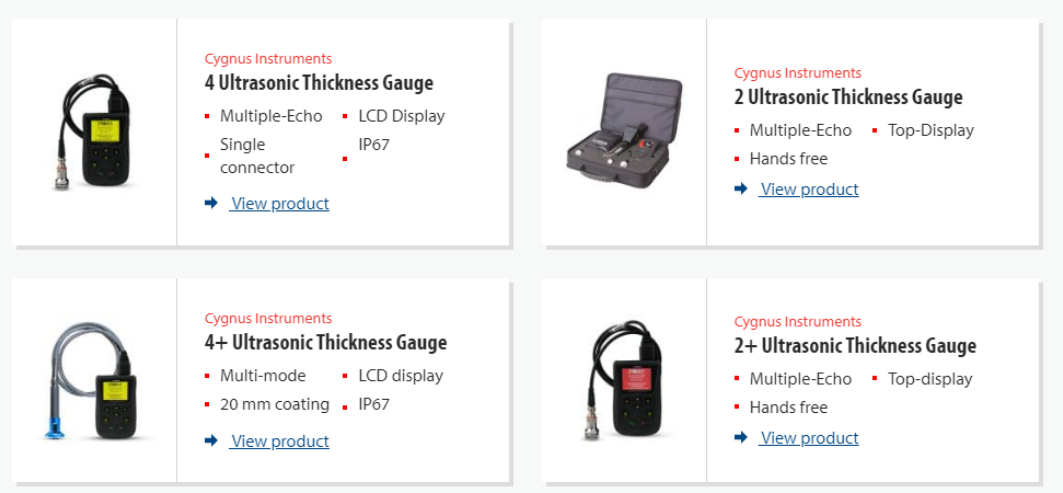


Рисунок 1.2 – Каталог компанії GE Measurement & Control

GE Measurement & Control виробляє широкий спектр продукції для контролю якості та безпеки виробництва. Це включає в себе апаратні засоби (наприклад, дефектоскопи, ультразвукові, рентгенівські пристрої) і програмне забезпечення для контролю матеріалів, деталей, конструкцій та систем у різних

галузях, таких як авіація, нафтогазова промисловість, автомобільна і суднобудівна промисловість, медичні технології та багато інших.

Компанія займає провідні позиції на ринку пристроїв для неруйнівного контролю завдяки своїм передовим технологіям, широкому спектру продукції і високій якості послуг. GE Measurement & Control активно розвиває інноваційні технології, такі як вдосконалені системи візуалізації даних, поліпшені алгоритми обробки сигналів та відображення результатів контролю для забезпечення найвищої точності та надійності даних. Їхній фокус на передових технологіях та інноваціях, широкий спектр продукції та глибоке розуміння потреб різних галузей виробництва роблять їх ключовим гравцем на ринку неруйнівного контролю та вимірювань. На рисунку 1.2 можна переглянути каталог виробів компанії, який налічує ультразвукові вимірювачі товщини.

1.2.2 Novotest

Компанія Novotest[5] – це вітчизняний виробник та постачальник високоякісного обладнання для неруйнівного контролю, вимірювань та тестування в різних сферах промисловості. Основний акцент у їхній діяльності робиться на розробці та виробництві пристроїв для вимірювання товщини, випробувань матеріалів, ультразвукового контролю та інших видів технічного контролю.

Їх продукція включає в себе широкий асортимент пристроїв, таких як ультразвукові дефектоскопи, вихрострумові дефектоскопи, товщиноміри, випробувальні машини та інше обладнання для якісного контролю продукції та матеріалів.

Українська компанія Novotest відома своєю високою якістю продукції та інноваційними рішеннями у сфері технічного контролю. Вони постійно вдосконалюють та модернізують свої пристрої, впроваджуючи нові технології та методи контролю, що відповідають найсучаснішим стандартам та вимогам ринку.

Однією з основних переваг Novotest є їхня конкурентоспроможна цінова політика, а також гнучкість у виготовленні обладнання, що дозволяє враховувати специфічні потреби клієнтів.

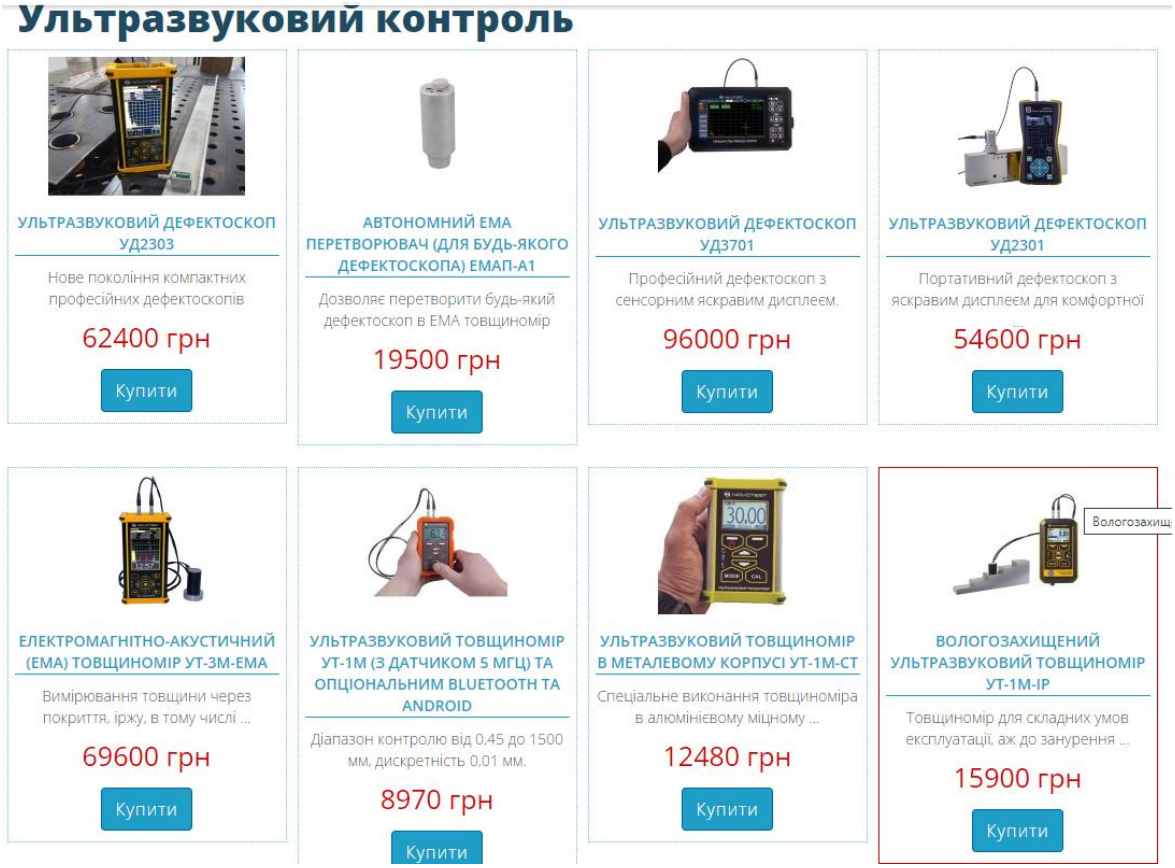


Рисунок 1.3 – Каталог компанії Novotest

Незважаючи на те, що Novotest може бути меншим за розміром порівняно з глобальними гравцями, компанія відома своєю спроможністю конкурувати на ринку завдяки високій якості продукції, інноваціям та гнучкості у відповіді на потреби своїх клієнтів. Ці прилади широко використовуються в промисловості для неруйнівного контролю якості матеріалів та структур, забезпечуючи надійність і безпеку виробничих процесів. На рисунку 1.3 можна переглянути каталог виробів компанії, який налічує ультразвукові вимірювачі товщини, дефектоскопи.

1.2.3 Olympus

Корпорація Olympus[6] — це відомий світовий виробник промислового обладнання та рішень у сфері наукових досліджень, медицини, індустрії та іншого. Їх продукція охоплює широкий спектр вимірювальних приладів, медичне обладнання, обладнання для неруйнівного контролю та інші технічні пристрої.

Однією з ключових галузей діяльності Olympus є виробництво вимірювальних приладів та обладнання для наукових досліджень та промисловості. Вони пропонують оптичне обладнання, мікроскопи, ендоскопи, ультразвукове обладнання, прилади для неруйнівного контролю матеріалів, включаючи вихрострумові, ультразвукові та рентгенівські системи, термокамери та інше.

Olympus займає провідні позиції на ринку завдяки своїм високоякісним технологіям та інноваціям. Їх продукція відома високою якістю зображення та точністю вимірювань, що робить їхні прилади та обладнання популярними серед спеціалістів у багатьох сферах.

Однією з переваг Olympus є їхня технологічна компетентність та досвід у багатьох галузях, що дозволяє їм розробляти інноваційні рішення для різних секторів промисловості та науки. Крім того, вони активно впроваджують нові технології, такі як цифрові системи обробки зображень та вдосконалені методи досліджень, що робить їхні продукти конкурентоспроможними на ринку.

1.2.3 Sonatest

Компанія Sonatest[7] є визнаним лідером у галузі ультразвукової неруйнівної дефектоскопії (НДТ). Заснована у Великобританії, ця компанія спеціалізується на розробці та виробництві високоякісних ультразвукових обладнань для перевірки матеріалів на дефекти. Її продукція використовується в багатьох галузях, включаючи авіацію, нафтогазову промисловість,

суднобудування та металургію. Основні пристрої та послуги яка надає компанія це:

- портативні ультразвукові дефектоскопи: ці прилади легкі, мають високу точність і зручні у використанні для мобільних застосувань;
- автоматизовані системи НДТ: для великомасштабних перевірок, таких як трубопроводи або великі металеві конструкції;
- програмне забезпечення для аналізу даних: дозволяє детально аналізувати дані, отримані від ультразвукових перевірок.

Sonatest займає міцну позицію на ринку НДТ обладнання завдяки своїй інноваційності та надійності продукції. Компанія є відомим постачальником обладнання для багатьох промислових гігантів, а також для малого та середнього бізнесу в цій галузі.

Sonatest завжди на передньому краї технологічних інновацій, регулярно оновлюючи свої продукти. Вироби відомі своєю високою якістю та надійністю, що забезпечує довгий термін служби приладів. Фірма приділяє велику увагу зручності користування своїх приладів, роблячи їх інтуїтивно зрозумілими та простими у використанні, надає післяпродажну підтримку і сервіс, що є важливим фактором для багатьох клієнтів.

Сумарно, Sonatest вирізняється на ринку завдяки своїм інноваціям, високій якості продукції, користувацькому досвіду та сильній післяпродажній підтримці.

Sonatest випускає широкий асортимент ультразвукового неруйнівного дефектоскопічного обладнання. Кожен продукт має унікальні характеристики та призначений для різних застосувань у галузі неруйнівного контролю. Ось кілька конкретних продуктів Sonatest:

Veo+: Це портативний ультразвуковий дефектоскоп, який використовується для точної ідентифікації дефектів у матеріалах. Veo+ володіє високою роздільною здатністю та гнучкістю налаштувань, ідеально підходить для складних застосувань, включаючи зварювальні шви, композитні матеріали та високотемпературні дослідження.

Prisma: Prisma – це ще один високопродуктивний ультразвуковий дефектоскоп, який особливо ефективний у забезпеченні точності виявлення дефектів і легкості використання. Цей прилад включає інтуїтивно зрозумілий інтерфейс із сенсорним екраном та великим вибором програмних опцій, що робить його гнучким для різних промислових застосувань. Прилад можна переглянути на рисунку 1.4.

Рисунок 1.4 – Ультразвуковий дефектоскоп Prisma

WheelProbe 2: Це ультразвуковий сканер, який використовується для перевірки плоских та криволінійних поверхонь. Його висока швидкість сканування та здатність адаптуватися до складних форм роблять його ідеальним для великомасштабних перевірок, таких як аерокосмічна та суднобудівна промисловість.

Sitescan D-50/D-70: Ці моделі є частиною лінійки Sitescan і представляють собою портативні дефектоскопи, які є чудовим вибором для загального використання. Вони відрізняються простотою використання, надійністю та доступністю, що робить їх популярним вибором у багатьох галузях. Прилад можна переглянути на рисунку 1.5.

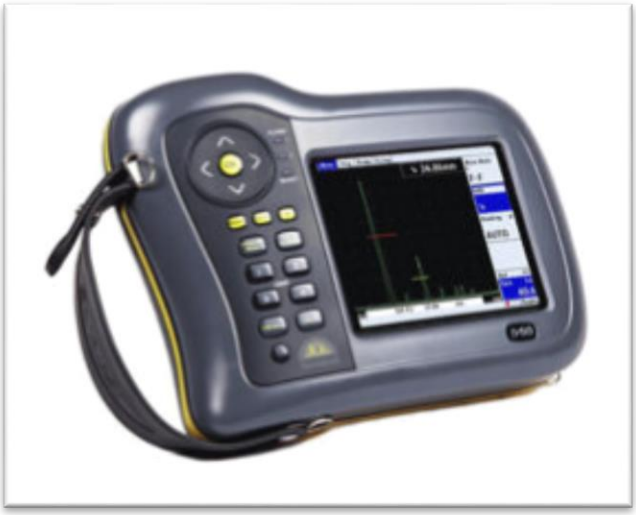


Рисунок 1.5 – Дефектоскоп Sitescan D-50

Masterscan D-70: Masterscan D-70 – це більш продвинутий дефектоскоп, який забезпечує високий рівень продуктивності та гнучкості. Він ідеально підходить для вимогливих застосувань, включаючи глибоке проникнення і детальну оцінку структурних дефектів. Прилад можна переглянути на рисунку 1.6.

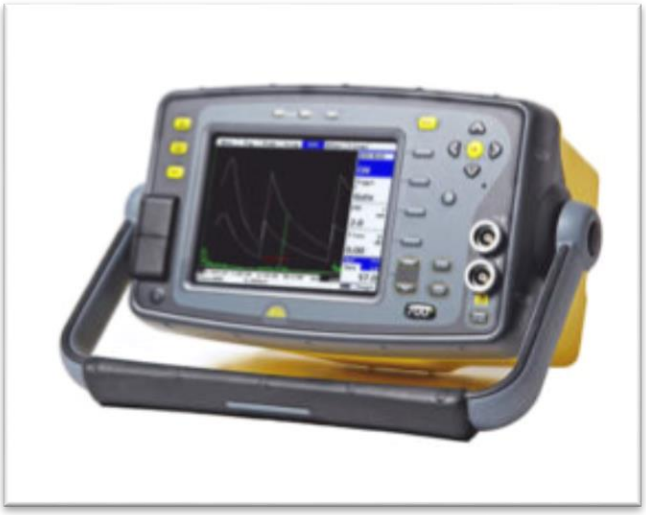


Рисунок 1.6 – Дефектоскоп Masterscan D-70

Для створення порівняльної таблиці ринку ультразвукових дефектоскопів, я розгляну кілька ключових брендів та їх продукти, включаючи Sonatest. Така таблиця може включати такі параметри як бренд, модель, ключові функції, застосування, та особливості. Оскільки я не маю доступу до останніх даних ринку, нижче наведена таблиця 1.1 буде заснована на загальнодоступній інформації та прикладах звичайних продуктів у цій галузі:

Таблиця 1.1 – Результати порівняння пристроїв

№	Бренд	Модель	Ключові Функції
1	2	3	4
1	Sonatest	Veo+	Висока роздільна здатність, гнучкі налаштування
2	GE	USM Go+	Легкий, міцний корпус, DAC/TCG і DGS функції
3	Olympus	EPOCH 650	Широкий діапазон частот, живлення від акумулятора
4	Krautkramer	USN 60	Цифрова регулювання звуку, автоматизований контроль
5	Proceq	Flaw Detector 100	Портативний, простий інтерфейс

Висновки до розділу 1

У першому розділі ми розглянули порівняльний аналіз різних моделей ультразвукових дефектоскопів від провідних виробників у галузі, таких як Sonatest, GE, Olympus, Krautkramer, та Proceq. Цей аналіз демонструє, що, хоча всі ці прилади мають загальну мету – виявлення дефектів у матеріалах, кожен з них має унікальні характеристики, які роблять його більш або менш підходящим для конкретних промислових застосувань. Хоча всі розглянуті дефектоскопи використовують ультразвукову технологію, вони мають різноманітні функції та можливості. Наприклад, деякі моделі акцентують на високій роздільній здатності та гнучкості налаштувань, тоді як інші зосереджуються на простоті використання та міцності. Різні моделі оптимізовані для різних типів застосувань. Наприклад, деякі призначені для важких умов експлуатації, як-от нафтогазова промисловість або авіація, тоді як інші ідеальні для детального аналізу в суднобудуванні або залізничній галузі. Ринок ультразвукових дефектоскопів є різноманітним і конкурентоспроможним, з кожним виробником, який пропонує унікальні переваги у своїх продуктах. Вибір конкретного обладнання залежить від специфічних потреб користувача, включаючи

Основним висновком цього аналізу є те, що серед розглянутих моделей ультразвукових дефектоскопів не було знайдено аналогів розробленого пристрою, основна відмінність якого полягає в зосередженні на всьому об'єкті, а не на детальній перевірці конкретної ділянки. Ця унікальна характеристика відкриває нові можливості у сфері неруйнівного контролю, дозволяючи користувачам проводити більш широкий огляд об'єктів без потреби зосереджуватися на окремих ділянках.

Це може бути особливо корисним у промислових застосуваннях, де потрібно швидко оцінити великі об'єкти або складні конструкції, забезпечуючи ефективність та економію часу. Такий підхід може значно відрізнятися від традиційних методів не руйнівного контролю, які часто зосереджені на

детальному скануванні окремих ділянок, що може бути часомістким та менш ефективним для великих об'єктів. Відсутність прямих аналогів цьому пристрою на ринку підкреслює його інноваційний потенціал та можливість задовольнити унікальні потреби користувачів, які шукають більш охоплюючі та швидкі рішення для неруйнівного контролю.

2 АПАРАТ ВИРІШЕННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ

2.1 Аналіз загальної архітектури

Пристрій для неруйнівного контролю можна концептуально розділити на дві основні частини: апаратну та програмну, можна розглянути детальніше на рисунку 2.1.

Апаратна частина пристрою базується на використанні контролера Arduino. Arduino – це відкрита платформа на основі простого мікроконтролера, що дозволяє легко інтегрувати різноманітні сенсори та актуатори, створюючи гнучкі та функціональні системи. У цій частині системи використовуються сенсори на основі п'єзоелементів, які відповідальні за реєстрацію коливань, що поширюються через металеву конструкцію об'єкта інфраструктури. Ці сенсори здатні виявляти найдрібніші зміни у вібраціях, що можуть вказувати на наявність дефектів.

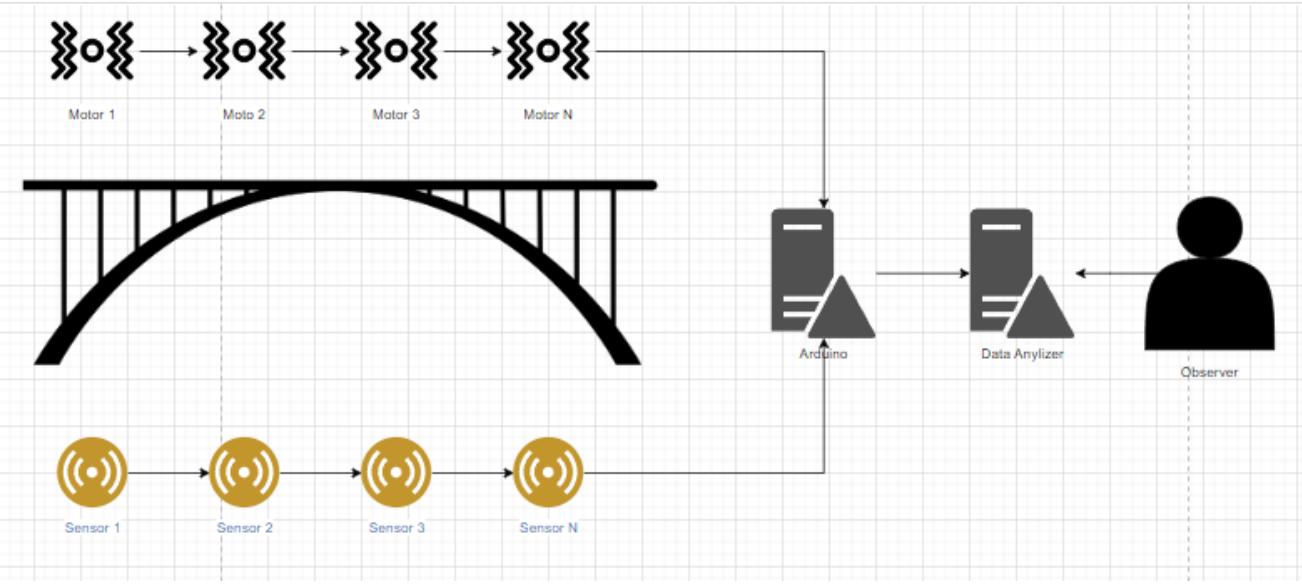


Рисунок 2.1 – Діаграма архітектури пристрою

До апаратної частини також входять збудники вібрацій, які створюють контрольовані вібрації в об'єкті. Ці збудники реалізовані за допомогою

електромоторів, які, обертаючись, генерують вібрації певної частоти та інтенсивності. Контролер Arduino координує весь процес, починаючи від активації електромоторів для генерації вібрацій, до збору даних з сенсорів. Отримана інформація зберігається і передається до програмної частини системи.

Програмна частина займається аналізом даних, отриманих від апаратної частини. Вона порівнює поточні дані з попередніми записами з цього ж об'єкту, щоб виявити будь-які відмінності в графіках та даних, які можуть вказувати на зміни в стані об'єкта. Програмне забезпечення також відіграє ключову роль у візуалізації даних для оператора, дозволяючи йому аналізувати перебіг тестування. У випадку виявлення критичних відмінностей, програма сповіщає оператора, що дозволяє своєчасно вжити необхідних заходів.

Ця інтеграція апаратної та програмної частини забезпечує комплексний підхід до неруйнівного контролю, дозволяючи ефективно виявляти та аналізувати потенційні дефекти в об'єктах інфраструктури.

2.2 Архітектура апаратної частини

Архітектура апаратної частини системи неруйнівного контролю сконцентрована на ефективному виявленні та моніторингу стану інфраструктурних об'єктів. Ця частина системи є фундаментом для збору даних, які потім аналізуються в програмній частині. Основні компоненти апаратної частини включають:

1) контролер Arduino: як серце системи, контролер Arduino координує роботу всіх підключених до нього компонентів. Він відповідає за активацію сенсорів і збудників вібрацій, а також за збір та передачу даних до програмної частини. Arduino вибрано через його гнучкість, доступність і легкість інтеграції з різними видами сенсорів та пристроїв;

2) сенсори на основі п'єзоелементів: ці сенсори використовуються для виявлення і реєстрації вібрацій та інших коливань, що поширюються через конструкцію об'єкта. Вони є ключовими для визначення дефектів у матеріалі;
3) збудники вібрації: збудники вібрацій, реалізовані за допомогою електромоторів, створюють контрольовані вібрації в об'єкті. Ці вібрації потім реєструються сенсорами, дозволяючи аналізувати внутрішній стан об'єкта;
4) інтерфейс для комунікації: апаратна частина включає інтерфейси для передачі даних до програмної частини, такі як USB, Wi-Fi, або інші бездротові технології. Це забезпечує надійний зв'язок між апаратною частиною та програмним забезпеченням для ефективної передачі даних.

У рамках цієї роботи було використано концепцію апаратно-програмної інтеграції, яка об'єднує фізичні компоненти, такі як контролер Arduino, сенсори на основі п'єзоелементів, та електромотори для генерації вібрацій, із програмним забезпеченням для обробки та аналізу даних. Також була використана архітектура системи реального часу, що дозволяє швидко обробляти зібрану інформацію та реагувати на неї. Ця система також має елементи розподіленої системи, де обробка даних відбувається як на апаратному рівні, так і на програмному.

2.2.1 Апаратно-програмна інтеграція

Апаратно-програмна інтеграція є фундаментальним елементом в створенні сучасних технічних систем, поєднуючи фізичні компоненти з розширеними програмними функціями для ефективного вирішення складних завдань. Цей процес знайшов широке застосування у вбудованих системах, робототехніці, та в індустрії Інтернету речей, де необхідна тісна взаємодія між апаратними компонентами та програмним забезпеченням.

У контексті розробленої системи неруйнівного контролю, апаратно-програмна інтеграція відіграє важливу роль, оскільки вона дозволяє точно керувати апаратною частиною, такою як сенсори та генератори вібрації, за допомогою програмного забезпечення. Це забезпечує гнучкість в налаштуваннях та контролі за операціями. Крім того, інтеграція забезпечує ефективний збір та обробку даних, що є критично важливим для виявлення дефектів. Можливість оперативно реагувати на зміни в даних в режимі реального часу дозволяє оператору швидко ідентифікувати потенційні проблеми.

Також, програмне забезпечення може бути адаптоване або оновлене для відповідності змінюваним вимогам чи умовам, що забезпечує тривалий термін служби та актуальність системи. Це полегшує діагностику та технічне обслуговування, роблячи систему більш надійною та ефективною в довгостроковій перспективі. Використання апаратно-програмної інтеграції у цій розробці є ключовим для досягнення високої точності, надійності та гнучкості в неруйнівному контролі об'єктів інфраструктури.

2.2.2 Система реального часу

Система реального часу – це особливий тип комп'ютерної системи, яка спроектована для обробки даних і реагування на зовнішні події в строго визначені часові рамки. Такі системи важливі в сценаріях, де відгук на події має відбуватися без затримок, оскільки будь-які затримки можуть мати серйозні наслідки.

Системи реального часу мають широке застосування в різних галузях. Наприклад, вони використовуються в промисловому керуванні, де важливо швидко реагувати на зміни у виробничих процесах. У медицині вони застосовуються для моніторингу пацієнтів, де швидка реакція на зміни стану здоров'я може бути життєво важливою. В автомобільній промисловості, системи реального часу використовуються для керування роботою двигунів та інших критичних систем у транспортних засобах.

У контексті розробки системи неруйнівного контролю, використання системи реального часу є необхідним для забезпечення швидкого збору та аналізу даних з сенсорів. Це дозволяє системі негайно реагувати на будь-які виявлені аномалії або зміни в стані об'єкта, що може вказувати на потенційні дефекти або небезпеки. Швидкий відгук системи на зібрані дані є критично важливим для забезпечення безпеки та ефективності процесу неруйнівного контролю.

2.2.3 Вбудована Система

Вбудована система в контексті розробки системи неруйнівного контролю є фундаментальною частиною, що відіграє важливу роль у виконанні специфічних завдань. Це компактний, високоспеціалізований комп'ютерний блок, який інтегрований безпосередньо в пристрій, забезпечуючи точне керування сенсорами і обробку даних. Такі системи характеризуються мінімізацією розміру та енергоспоживання, водночас забезпечуючи високу продуктивність та надійність, що є ключовими факторами в нашій розробці.

У розробленій нами системі неруйнівного контролю, вбудована система, така як контролер Arduino, відповідає за точне виконання ряду критичних задач. Вона забезпечує керування сенсорами, які реєструють вібрації та інші важливі параметри об'єкту інфраструктури, а також за обробку зібраних даних для їх подальшого аналізу. Це не лише підвищує точність і надійність системи, але й забезпечує ефективне використання енергії та компактність, що є важливим для портативних систем неруйнівного контролю.

2.3 Архітектура програмної частини

Для системи неруйнівного контролю, програмна частина має відігравати критичну роль у обробці великих обсягів даних, їх аналізі, візуальному

представленні для оператора та виконанні сповіщень у разі виявлення дефектів. Ефективна архітектура програмної частини повинна включати наступні елементи:

- модуль збору даних: цей модуль відповідає за прийом даних з апаратної частини. він має забезпечувати стабільну та ефективну передачу даних від сенсорів до системи обробки;
- модуль обробки даних: основний блок, що займається аналізом зібраних даних. він включає алгоритми для виявлення відхилень, аномалій та потенційних дефектів, заснованих на заздалегідь визначених метриках та порівнянні з історичними даними;
- база даних: важливою частиною архітектури є система баз даних для зберігання зібраних даних та історичних записів. це дозволяє проводити порівняльний аналіз і виявляти тенденції або зміни у поведінці об'єкта протягом часу;
- інтерфейс користувача (UI): для ефективного візуального представлення даних оператору система повинна містити зручний і інтуїтивно зрозумілий користувацький інтерфейс. це включає графіки, діаграми та інші візуальні засоби для демонстрації стану об'єкта і результатів аналізу;
- система попереджень і сповіщень: для інформування оператора про критичні відхилення або потенційні дефекти важливо мати модуль сповіщень, який забезпечує своєчасну реакцію на виявлені проблеми;
- модуль звітності та аналітики: додатковий компонент, який дозволяє генерувати звіти та проводити глибокий аналіз зібраних даних для виявлення трендів та шаблонів.

Така архітектура забезпечує комплексний підхід до обробки, аналізу та представлення даних, дозволяючи операторам ефективно моніторити стан об'єктів інфраструктури та вчасно реагувати на будь-які виявлені проблеми.

Під час розробки системи неруйнівного контролю було використано модульну архітектуру, яка дозволила реалізувати широкий функціонал системи.

Від аналізу даних, отриманих від сенсорів, до сповіщення оператора про потенційні дефекти або відхилення – кожен аспект обробки та управління даними був ретельно спланований та розроблений у вигляді окремих модулів. Ця структура забезпечила високий рівень адаптивності та масштабованості, дозволяючи легко модифікувати або розширювати систему залежно від змінюваних потреб та умов експлуатації.

Крім того, для інтерфейсу програми була застосована клієнт-серверна архітектура. Це означає, що вся обробка та аналіз даних виконується на сервері, тоді як кінцевий користувач взаємодіє з системою через клієнтський інтерфейс. Така архітектура не тільки спрощує управління та підтримку системи, але й забезпечує безпеку та централізоване управління даними. Клієнтська частина дозволяє операторам легко візуалізувати та інтерпретувати результати аналізу, отримувати сповіщення про стан системи, а також здійснювати контроль та налаштування в режимі реального часу.

2.3.1 Модульна архітектура

Модульна архітектура – це підхід у проектуванні програмного забезпечення, де великі системи розбиваються на окремі, незалежні блоки, відомі як модулі. Кожен модуль виконує певну функцію і взаємодіє з іншими модулями через визначені інтерфейси. Основні сфери використання такого принципу архітектури:

- веб-додаток: модульний веб-додаток може мати окремі модулі для користувацького інтерфейсу, обробки даних, зв'язку з базою даних тощо;
- мобільний додаток: мобільний додаток може мати модулі для геолокації, платежів, чату, та інші.

У розробці системи неруйнівного контролю модульна архітектура була використана для забезпечення гнучкості, масштабованості та ефективності.

Розподіл функціоналу по модулях дозволяє легко вносити зміни або оновлення в окремі частини системи без впливу на інші модулі. Наприклад, модуль обробки даних може бути оновлений для використання нових алгоритмів без зміни інтерфейсу користувача. Модульна структура дозволяє легко додавати нові функції. Наприклад, можна додати модуль для інтеграції з додатковим обладнанням або системами без переробки існуючої системи. Кожен модуль можна тестувати окремо, що спрощує процес виявлення помилок та їх усунення. Такий підхід також полегшує процес технічного обслуговування системи. Тому серед основних переваг виділяють такі:

- гнучкість у розробці: головною перевагою модульної архітектури є її гнучкість у розробці. Розробники можуть зосередитися на окремих модулях, вносячи зміни та оновлення без ризику впливу на інші частини системи. Це сприяє більш швидкій та адаптивній розробці;
- легкість масштабування: модульна архітектура дозволяє системі ефективно масштабуватися, додаючи нові функціональні можливості або покращуючи існуючі. Це полегшує адаптацію системи до змінюваних потреб та забезпечує її тривалий розвиток;
- висока підтримуваність: кожен окремий модуль легше підтримувати та оновлювати, що знижує загальні витрати на обслуговування та утримання системи. Це також сприяє швидшому виправленню помилок та вдосконаленню функціональності;
- зниження залежностей: модульна структура мінімізує взаємозалежності між різними частинами системи, що знижує ризик помилок та конфліктів, які можуть виникнути при внесенні змін або оновленнях;
- спрощення тестування: окреме тестування кожного модуля дозволяє більш ефективно виявляти та усувати помилки, підвищуючи загальну надійність системи. Це спрощує тестування та забезпечує більш стабільну роботу системи в цілому.

Проте так само так архітектура має деякі недоліки:

- складність управління залежностями: у модульній архітектурі, хоча модулі розробляються незалежно, вони часто мають залежності між собою, що може ускладнювати управління та синхронізацію. Велика кількість залежностей між модулями може призвести до складнощів у розумінні взаємодії між ними та ускладнити процес внесення змін;
- витрати на інтеграцію: інтеграція різних модулів може вимагати значних витрат часу та ресурсів. Особливо це стосується випадків, коли модулі використовують різні технології або розроблені різними командами. Це може призвести до необхідності розробки додаткових інтерфейсів для забезпечення взаємодії між модулями;
- ризик неконсистентності даних: у системах, де дані розподілені між модулями, існує ризик неконсистентності даних, особливо якщо не існує централізованого механізму управління даними. Це може призвести до помилок та неузгодженості в результатах обробки даних;
- складнощі у тестуванні інтеграції: хоча окремі модулі можуть бути тестовані легко, інтеграційне тестування великої системи може бути дуже складним. Це вимагає перевірки взаємодії між усіма модулями, що може бути часомістким і складним процесом, особливо якщо модулі мають складні взаємозалежності;
- витрати на початкове планування: ефективна модульна архітектура вимагає ретельного планування та проектування на початкових етапах розробки. Це може збільшити витрати та час, необхідний для розробки, оскільки потрібно детально спланувати, як модулі будуть взаємодіяти та інтегруватися один з одним.

Вибір модульної архітектури замість інших підходів був обумовлений потребою у високій адаптивності та здатності системи швидко реагувати на змінювані умови та вимоги. В порівнянні з монолітними або тісно інтегрованими системами, модульна архітектура пропонує кращу підтримку інновацій та дозволяє більш ефективно.

2.3.2 Клієнт-серверна Архітектура

Клієнт-серверна архітектура є однією з основних моделей організації та взаємодії в мережевих системах. В цій моделі, процес обробки даних та управління ресурсами розділений між двома видами вузлів: серверами, які надають ресурси або послуги, та клієнтами, які ці ресурси використовують.
Клієнт-серверна архітектура є популярною моделлю взаємодії в мережевих системах, де функціонал та обробка даних розділені між двома основними компонентами: клієнтом та сервером.

У цій архітектурі, сервер є потужним вузлом, який надає ресурси або послуги (наприклад, веб-сторінки, бази даних, обчислювальні можливості), тоді як клієнт є користувацьким вузлом, який запитує та використовує ці ресурси. Ця модель заснована на принципі запиту та відповіді, де клієнт ініціює запит, а сервер обробляє його та надсилає відповідь.

Основною перевагою клієнт-серверної архітектури є її централізована структура. Сервери є відповідальними за управління центральними ресурсами, що спрощує процеси зберігання даних, управління доступом та безпекою. Це дозволяє легше масштабувати систему, управляти навантаженнями та забезпечувати високий рівень ефективності обслуговування великої кількості клієнтів. З іншого боку, клієнти можуть бути легкими та мобільними, надаючи кінцевим користувачам зручний доступ до ресурсів сервера через різноманітні інтерфейси.

Важливим аспектом клієнт-серверної архітектури є мережеве з'єднання між клієнтом та сервером, яке може бути реалізовано через різні технології (наприклад, через Інтернет або локальну мережу). Ця модель надає гнучкість у виборі платформ та технологій, дозволяючи розробникам оптимізувати обидві сторони системи для різних завдань – від простого веб-браузінгу до складних розподілених обчислень та великих баз даних. В результаті, клієнт-серверна архітектура є фундаментом для багатьох сучасних бізнес-додатків та Інтернет-

сервісів, забезпечуючи їх ефективність, масштабованість та доступність. На рисунку 2.2 зображено діаграму клієнт-серверної архітектури.

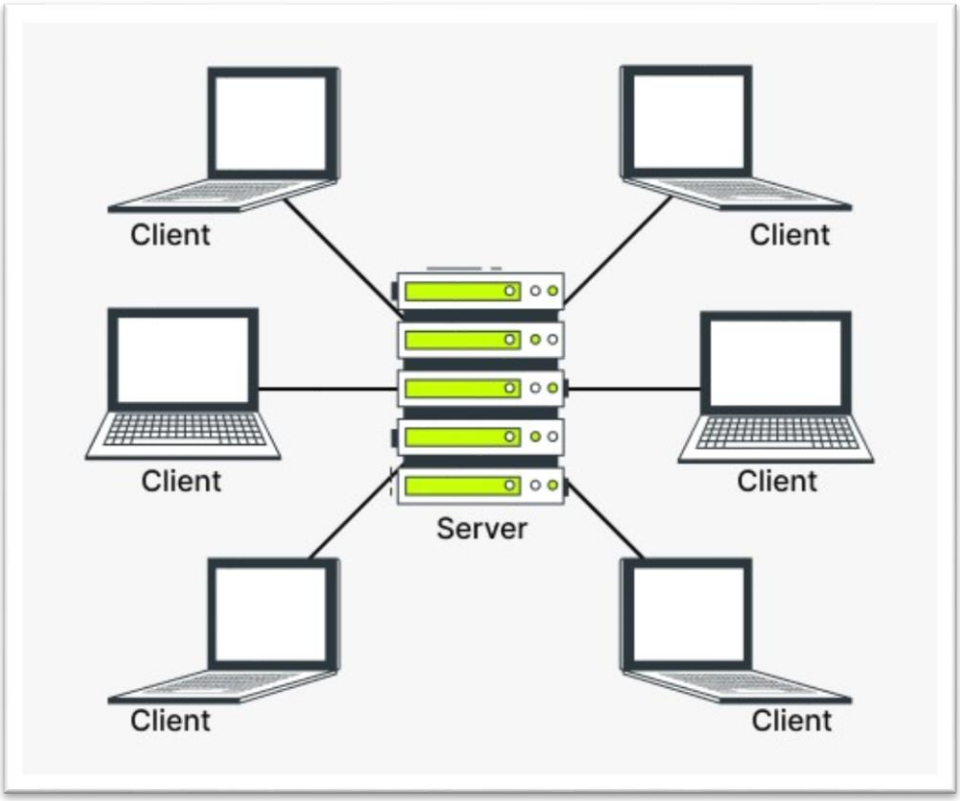


Рисунок 2.2 – Клієнт-серверної архітектура

Клієнт-серверна архітектура порівнюється з альтернативними моделями, такими як Peer-to-Peer (P2P) зображений на Рисунку 2.3 та Client-Queue-Client зображений на Рисунку 2.4. У клієнт-серверній моделі вся міць зосереджена на сервері, тоді як P2P розподіляє функції та ресурси між всіма вузлами, виконуючи ролі як клієнтів, так і серверів. Client-Queue-Client архітектура використовує пасивні черги для спрощення взаємодії між клієнтами та серверами.

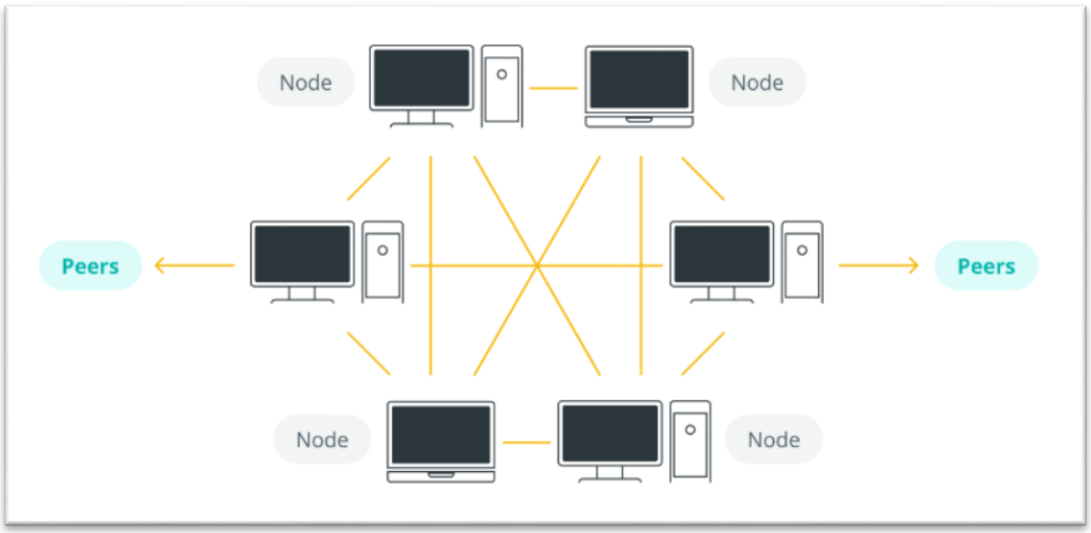


Рисунок 2.3 – Peer-to-Peer архітектура

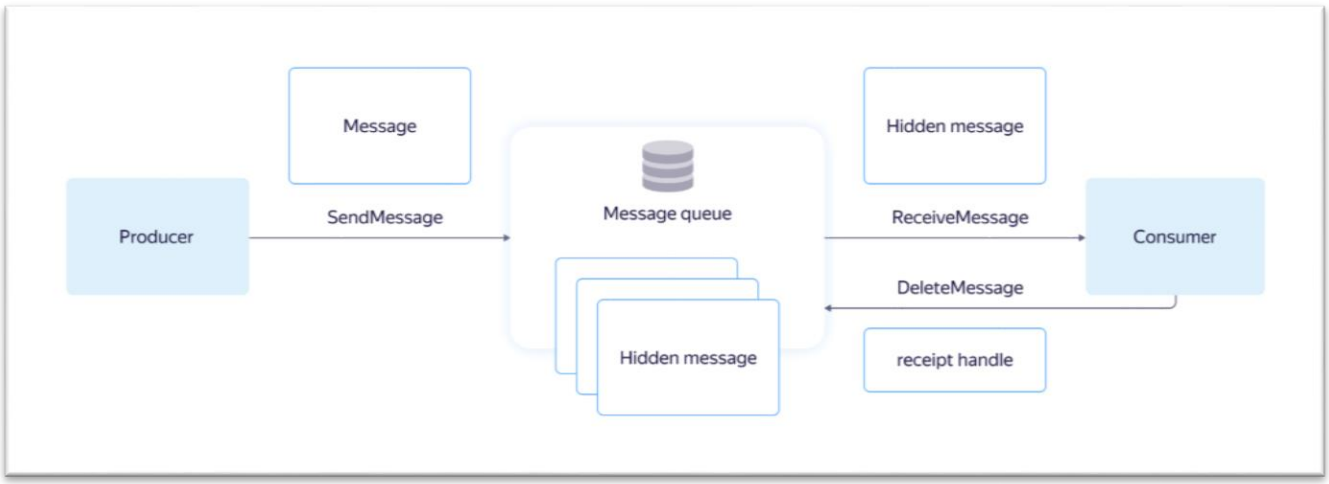


Рисунок 2.4 – Client-Queue-Client архітектура

Основними перевагами клієнт-серверної архітектура називають:

- централізоване управління ресурсами: ключовою перевагою клієнт-серверної архітектури є централізоване управління ресурсами. Сервер є головним вузлом, що забезпечує обробку даних, зберігання та доступ до спільних ресурсів. Це значно спрощує управління великими обсягами даних та забезпечує їх консистентність;
- легше масштабування: завдяки централізованому характеру сервера, систему можна легко масштабувати, додавши більше серверів або

оновивши існуючі, що є ефективним рішенням для зростаючого навантаження або розширення функціональності;

- поліпшена безпека: контроль доступу та захист даних на сервері дозволяє ефективно забезпечувати безпеку інформації. Централізація даних усуває необхідність розподілення чутливої інформації по мережі, знижуючи ризик несанкціонованого доступу.
- ефективність мережевого трафіку: оскільки основна обробка даних відбувається на сервері, це знижує навантаження на мережу та покращує її ефективність. Клієнти відправляють запити та отримують відповіді, але не займаються інтенсивною обробкою даних, що знижує загальне навантаження на мережу;
- спрощене технічне обслуговування та оновлення: управління та оновлення системи в клієнт-серверній архітектурі здійснюється переважно на сервері, що спрощує процеси технічного обслуговування та оновлення. Зміни на сервері автоматично застосовуються до всіх клієнтів, що зменшує потребу в індивідуальному оновленні кожного клієнтського пристрою.

Переваги клієнт-серверної архітектури включають легше управління, безпеку та адміністрування. Оскільки дані зосереджені на сервері, оновлення та управління даними здійснюються централізовано. Це забезпечує ефективність у великих мережевих середовищах. Водночас, P2P архітектура має перевагу у розподілі ресурсів та забезпечує кращу масштабованість і відновлення при відмовах окремих вузлів. Однак, управління та безпека в P2P можуть бути складнішими через розподіл ресурсів і відсутність централізованого контролю. Виділять такі недоліки клієнт серверної архітектури:

- централізована точка відмови: основним недоліком клієнт-серверної архітектури є те, що вона має одну централізовану точку відмови – сервер. Якщо сервер зазнає збою або виходить з ладу, це може призвести

до повної непрацездатності всієї системи, оскільки всі клієнти залежать від його роботи;

- масштабування сервера: підтримання зростаючих потреб системи може вимагати значного масштабування сервера. Це може бути складно та вимагати значних витрат, особливо якщо потрібно додавати апаратне забезпечення або модернізувати існуюче;
- проблеми з безпекою: через централізацію даних, сервери є привабливими цілями для кібератак. Це вимагає високого рівня безпеки та постійного моніторингу, що може бути складним і ресурсоємним;
- навантаження на сервер: всі запити від клієнтів обробляються сервером, що може призвести до його перевантаження. При великій кількості одночасних запитів, це може вплинути на швидкість відповідей та загальну продуктивність системи;
- затрати на інфраструктуру та управління: встановлення та утримання серверів може бути дорогим. Потужні сервери вимагають значних інвестицій, а також кваліфікованого персоналу для управління та підтримки серверної інфраструктури, що збільшує загальні витрати на утримання системи.

Клієнт-серверна архітектура була вибрана для проекту з декількох причин. По-перше, вона забезпечує централізоване управління та обробку даних, що є важливим для великих обсягів інформації. По-друге, ця архітектура дозволяє легко масштабувати систему та ефективно управляти ресурсами. Також, клієнт-серверна модель сприяє кращій безпеці, оскільки всі дані централізовано зберігаються на сервері, що полегшує контроль доступу та захисту даних. Все це сприяло високій ефективності, надійності та гнучкості системи, що є критичним для успішної реалізації проекту.

2.4 Технології апаратної частини

Розглянемо детально технології (мови програмування, бібліотеки, контроллери та компонети), які використовуються для апаратної частини проекту.

2.4.1 Arduino

Arduino[8] є відкритою апаратною платформою, яка стала популярною завдяки своїй доступності та легкості використання. Вона заснована на простих мікроконтролерах і включає серію плат, які можуть бути програмовані через спеціалізоване розробницьке середовище. Arduino підходить для широкого спектру проектів, від освітніх до хобі-рівня, і навіть в промислових застосуваннях, на рисунку 2.5 можна побачити приклад контролеру ардуіно.

Рисунок 2.5 – Контролер Arduino

Завдяки своїй гнучкості, Arduino дозволяє реалізовувати складні проекти з використанням датчиків, моторів, світлодіодів та інших компонентів. Його програмування засноване на C/C++-подібному мові, що робить його доступним для широкої аудиторії. Спільнота Arduino розвинула численні бібліотеки та підтримує активний обмін знаннями, що сприяє поширенню інноваційних ідей та розвитку нових проектів.

Для проекту було використана версія Arduino Uno, яка є фундаментальним вибором для багатьох проектів на основі Arduino завдяки своїм універсальним характеристикам. Arduino Uno оснащений мікроконтролером ATmega328P та має 32 КБ флеш-пам'яті (з яких 0.5 КБ використовується для завантажувача), 2 КБ SRAM і 1 КБ EEPROM. Щодо підключень, Uno має 14 цифрових вхідно-вихідних пінів, включаючи 6 аналогових входів і 6 PWM пінів. Додатково, плата має USB-інтерфейс для програмування та взаємодії з комп'ютерами, а також зовнішній роз'єм живлення. Її робоча напруга становить 5 В, а рекомендована вхідна напруга – від 7 до 12 В. Плата Arduino Uno підтримує I2C та SPI комунікації, що робить її підходящою для підключення різноманітних датчиків та актуаторів. Вона оснащена вбудованим LED-індикатором на піні 13, який може використовуватися для базового відлагодження коду. Плата також має зручні заголовки для підключення додаткових модулів та датчиків. Крім того, Arduino Uno сумісна з різними розширювальними платами (shields), які легко встановлюються на плату та розширюють її функціональність.

Arduino Uno ідеально підходить для освітніх цілей, хобі-проектів та навіть деяких професійних застосувань, завдяки своїй відкритій платформі та широкій підтримці з боку спільноти, але спільнота також виділяє деяки недоліки такої платформи, ці недоліки важливо враховувати при виборі платформи для конкретного проекту.

Одним із ключових недоліків платформи Arduino Uno є її обмежена обчислювальна потужність та пам'ять. З ATmega328P мікроконтролером, Uno має лише 2 КБ SRAM, 32 КБ флеш-пам'яті (з яких частина зайнята завантажувачем),

та 1 КБ EEPROM. Це може не вистачати для більш вимогливих задач, таких як обробка великих даних, складні обчислювальні процеси чи робота з ресурсоємними бібліотеками. Такі обмеження роблять Arduino менш підходящою для розширених проектів або додатків, що вимагають більшої пам'яті та обчислювальної потужності.

Arduino Uno не має вбудованого Інтернет-з'єднання, що означає відсутність нативної підтримки Wi-Fi або Ethernet без зовнішніх модулів. Це обмежує можливості Arduino в проектах, де необхідне пряме підключення до Інтернету для таких цілей, як збір даних, віддалене керування або Інтернет речей (IoT). Щоб забезпечити з'єднання з Інтернетом, користувачам доводиться використовувати додаткові модулі, такі як Wi-Fi шилди, що вимагає додаткових витрат та комплектації.

Arduino Uno не має вбудованого захисту від перенапруги, що може стати проблемою при неправильному використанні або в умовах сильних електричних перешкод. Це робить її вразливою до можливих пошкоджень від високого напруги або струму, які можуть випадково бути підключені до плати. В результаті, користувачам необхідно бути особливо уважними при підключенні джерел живлення або інших компонентів, щоб уникнути перенапруги, яка може пошкодити Arduino Uno.

Оскільки Arduino дуже популярна система, то у неї є дуже вагомі переваги. Першою значною перевагою платформи Arduino Uno є її доступність та низька вартість. Ця плата мікроконтролера доступна за ціною, яка робить її привабливою для широкого кола користувачів – від студентів та хобістів до освітніх закладів та дослідників. Низька ціна сприяє експериментуванню та інноваціям, дозволяючи користувачам реалізувати різноманітні проекти без значних фінансових витрат. Окрім цього, доступність Arduino Uno сприяє її популярності в освітніх програмах, де важливою є вартість обладнання;

Другою перевагою Arduino Uno є її простота використання, що робить її ідеальною для початківців у світі програмування та електроніки. Її програмне

забезпечення засноване на простих та інтуїтивно зрозумілих принципах, що дозволяє новачкам швидко освоїти основи програмування та роботи з електронними компонентами. Використання зрозумілої мови програмування, такої як C/C++, та наявність великої кількості готових до використання бібліотек і кодів спрощує процес розробки та експериментування, навіть для тих, хто тільки розпочинає знайомство з електронікою;

Третьою перевагою Arduino Uno є її гнучкість, яка робить її придатною для широкого спектру проектів. Завдяки 14 цифровим вхідно-вихідним пінам, включаючи 6 пінів для PWM (широтно-імпульсної модуляції), та 6 аналогових входів, Arduino Uno може інтерфейсуватися з різними сенсорами, актуаторами, дисплеями та іншими електронними компонентами. Це відкриває можливості для реалізації різноманітних електронних проектів, від простих до більш складних;

Arduino Uno є наявність активної та великої спільноти користувачів і розробників. Ця спільнота надає значну підтримку у вигляді онлайн форумів, навчальних матеріалів, детальних підручників та багатої колекції бібліотек коду. Це не лише сприяє спрощеному навчанню та швидкому вирішенню проблем, але й стимулює обмін ідеями та розвиток інноваційних проектів. Доступність ресурсів та знань від спільноти робить Arduino Uno особливо привабливим для тих, хто хоче швидко втілити свої творчі ідеї в життя;

Сумісність Arduino Uno з розширювальними платами (shields) значно розширює її можливості. Ці плати дозволяють додавати додаткові функції, такі як підключення до Інтернету, керування моторами, GPS та багато іншого, без потреби в складному електронному дизайні. Це робить Uno гнучкою платформою, яка може адаптуватися до широкого спектру проектів.

Враховуючи переваги Arduino, такі як доступність, простота використання, гнучкість, підтримка спільноти та сумісність з розширювальними платами, та обмеження, включаючи обмежену обчислювальну потужність та відсутність вбудованого Інтернет-з'єднання, Arduino Uno було вибрано для проекту через його оптимальне співвідношення функціональності, вартості та простоти

використання. Це робить її ідеальною для широкого спектру проектів, особливо для освітніх, хобі-проектів та прототипування, такий як мій проект, якому було необхідне просте рішення для вирішення апаратних задач.

2.4.2 П'єзоелектричні датчики вібрації

П'єзоелектричні датчики вібрації – це пристрої, що використовують п'єзоелектричний ефект для виявлення і вимірювання вібрацій або механічних змін. Вони перетворюють механічні вібрації на електричний сигнал. П'єзоелектричні датчики вібрації відрізняються високою чутливістю та здатністю точно вимірювати вібрації навіть у складних умовах, наприклад, у високотемпературних або вологих середовищах. Вони зазвичай мають компактний розмір, що робить їх ідеальними для вбудовування в різноманітне обладнання. Ці датчики широко застосовуються для моніторингу стану машин, діагностики дефектів, в автоматизованих системах контролю та у тестуванні матеріалів[9].

Ці датчики ефективні у високочастотному діапазоні та забезпечують високу точність, роблячи їх ідеальними для застосування в промисловості, автомобільній галузі та інженерії, приклад такого датчику можна переглянути на рисунку 2.6.

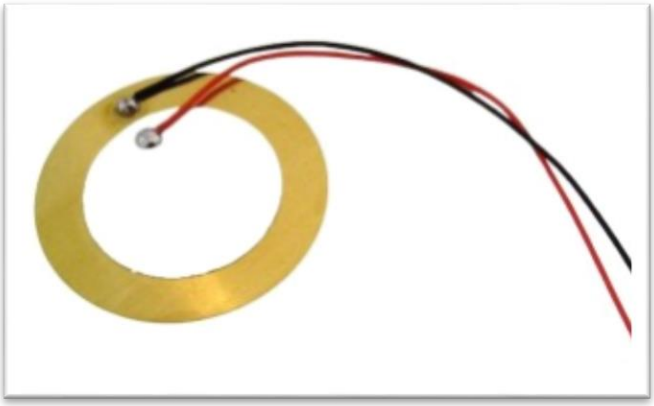


Рисунок 2.6 – П'єзоелектричний датчик вібрацій

При виконанні роботи виникали сумніви щодо вибору між п'єзоелектричними датчиками вібрації та тензорезистивними датчиками. Обидва типи датчиків мають свої переваги та обмеження, що ускладнює вибір. П'єзоелектричні датчики підходять для вимірювання вібрацій і мають високу точність, тоді як тензорезистивні датчики краще підходять для вимірювання тиску або натягу[10, 11].

Тензорні датчики, або тензометричні датчики, – це пристрої, які використовуються для вимірювання напруги, деформацій або тиску. Вони перетворюють механічне навантаження (натяг, стиск або згин) на електричний сигнал. Ці датчики зазвичай використовуються у вагових системах, конструкційних випробуваннях та в промислових застосуваннях, де важливо точно вимірювати механічні зміни. Тензорні датчики, зображений на рисунку 2.7, використовують принципи тензометрії, використовуючи зміну електричного опору під час деформації матеріалу, для перетворення механічного навантаження у електричний сигнал. Ці датчики знаходять широке застосування в різних галузях, включаючи автомобільну промисловість, аерокосмічну галузь та цивільне будівництво, забезпечуючи точність та надійність у вимірюваннях.

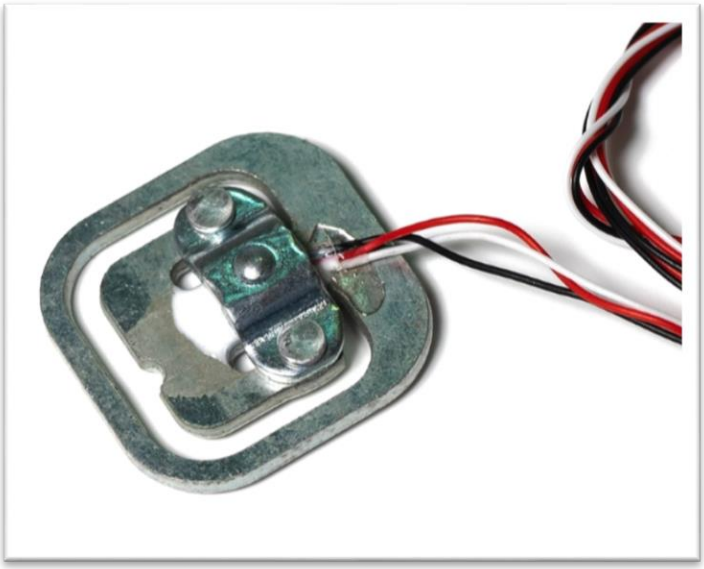


Рисунок 2.7 – П'єзоелектричний датчик вібрацій

На основі переваг і недоліків було зроблено таблицю 2.1 для порівняння:

Таблиця 2.1 – Порівняння видів датчиків

№	Критерій	Тензорні Датчики	П'єзоелектричні Датчики
1	Вимірювання	Напруга, деформація, тиск	Вібрації, зміни тиску
2	Принцип роботи	Зміна електричного опору	Генерація електричного заряду
3	Застосування	Вагові системи, конструкційні випробування	Високочастотні вимірювання, моніторинг вібрацій
4	Галузі	Будівництво, автомобільна промисловість	Автомобільна галузь, медичні прилади, робототехніка

Після ретельного аналізу переваг та недоліків обох типів датчиків було вирішено використовувати п'єзоелектричні датчики для проекту. Це рішення було засноване на їх здатності точно вимірювати вібрації, високій чутливості та надійності в складних умовах. Ці особливості роблять п'єзоелектричні датчики оптимальними для проекту, де важливі точність вимірювань та стабільність роботи.

2.4.3 Генератор вібрацій

Генератори вібрацій – це пристрої, які використовуються для створення та контролю вібрацій. Вони можуть варіювати за розміром та потужністю, пропонуючи різні рівні вібрацій для широкого спектру застосувань, включаючи тестування матеріалів, випробування конструкцій, наукові дослідження та

промислове виробництво. Ці пристрої можуть генерувати вібрації з точно контрольованою амплітудою, частотою та тривалістю, що є важливим для точних та надійних вимірювань.

Генератори вібрацій бувають різних типів, залежно від їх застосування та характеристик:

- електродинамічні генератори вібрацій широко використовуються для точних випробувань та аналізу вібрацій. Ці пристрої здатні створювати вібрації з високою частотою та контрольованою амплітудою, що робить їх ідеальними для лабораторних тестів та промислового використання. Вони часто застосовуються у секторах, де необхідно точно відтворити умови експлуатації обладнання та випробувати його на стійкість до вібраційних навантажень;
- електромагнітні генератори вібрацій створюють вібрації за допомогою електромагнітних полів. Вони зазвичай використовуються для створення вібрацій з низькою до середньою частотою. Цей тип генераторів підходить для застосувань, де потрібна менш інтенсивна вібрація. Вони ефективні для використання в ситуаціях, де потрібно симулювати реальні умови навантаження або перевіряти механічну стійкість продуктів і матеріалів;
- п'єзоелектричні генератори вібрацій: п'єзоелектричні генератори вібрацій використовують п'єзоелектричний матеріал для генерації вібрацій, коли на нього надається електрична напруга. Ці генератори дуже точні та ефективні для створення дрібних, високочастотних вібрацій. Їх широко використовують у наукових дослідженнях, високоточному обладнанні та у медичних приладах, де потрібні надзвичайно точні та контрольовані вібрації, приклад якого можна побачити на рисунку 2.8;
- електромеханічні генератори вібрацій зазвичай використовують електромотори для створення механічних вібрацій, приклад якого можна

побачити на рисунку 2.9. Ці пристрої мають здатність створювати вібрації в широкому діапазоні частот і амплітуд, що робить їх підходящими для різних застосувань, включаючи тестування міцності продуктів, вібраційні випробування у промисловості та наукові дослідження. Вони можуть бути особливо корисними у випадках, коли потрібно імітувати реальні умови експлуатації обладнання або конструкцій.

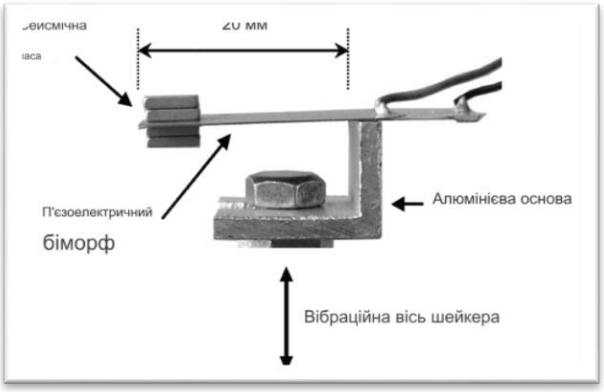


Рисунок 2.8 – П'єзоелектричний генератор вібрацій

Рисунок 2.9 – Електромеханічний генератор вібрацій

Після проведення досліджень і порівняння характеристик кожного виду з генераторів, був обраний електромеханічний генератор вібрацій. Вибір електромеханічного генератора вібрацій у вигляді маленького моторчика, подібного до тих, що використовуються у телефонах, можна пояснити кількома факторами. Такі моторчики компактні, легкі в інтеграції з іншими компонентами і відносно недорогі, що робить їх ідеальними для масового виробництва та широкого застосування. Вони здатні генерувати достатню вібрацію для багатьох застосувань, включаючи випробування та застосування для виявлення дефектів[12].

2.5 Технології програмної частини

Розглянемо детально технології (мови програмування, бібліотеки, фреймворки та інше), які використовуються для програмної частини проекту.

2.5.1 Платформа .NET

Платформа .NET була розроблена Microsoft і вперше представлена у 2002 році. Вона призначена для створення та запуску програмного забезпечення на різних платформах. Основні компоненти .NET включають Common Language Runtime (CLR) для виконання коду та обширну бібліотеку класів, що забезпечують функціональність для різних застосувань. Архітектура .NET підтримує багатомовність і крос-платформність, дозволяючи розробникам використовувати різні мови програмування та створювати програми, сумісні з різними операційними системами[13][14].

З часом .NET розвивалася, ставши більш гнучкою та крос-платформною з запуском .NET Core, який розширив можливості платформи за межі Windows.

Значним кроком у розвитку стало введення .NET 5 та вище, які об'єднали .NET Framework та .NET Core в єдину платформу. Це сприяло створенню єдиної бази коду для різних платформ, включаючи macOS, Linux та Windows. Основні архітектурні елементи, такі як CLR, бібліотеки класів та мови програмування, залишаються ключовими у структурі .NET, надаючи розробникам великий набір інструментів для реалізації різноманітних проектів. Отже розробники виділяють ключеві переваги, через які ця платформа була обрана для розробки проекту:

- багатомовність: .NET підтримує різноманітні мови програмування, такі як C#, VB.NET, і F#. Це надає розробникам свободу вибору мови, яка найкраще відповідає конкретному проекту, дозволяючи їм використовувати найсучасніші мовні особливості та парадигми;
- крос-платформність: з появою .NET Core, .NET забезпечує підтримку розробки додатків, які можуть працювати на різних операційних системах, включаючи Windows, Linux та macOS. Це розширює можливості розробників у створенні програм, що можуть обслуговувати більший спектр користувачів;
- масштабованість та висока продуктивність: .NET оптимізований для роботи як з малими, так і з великими масштабованими додатками, дозволяючи ефективно розподіляти ресурси та обробляти великі обсяги даних;
- сильні безпека та управління пам'яттю: завдяки вбудованим механізмам безпеки та автоматичному управлінню пам'яттю .NET знижує ризик витоку пам'яті та інших поширених проблем безпеки;
- розширена бібліотека класів: велика колекція попередньо написаних класів і бібліотек значно спрощує процес розробки, забезпечуючи широкий спектр функціональності «з коробки».

Платформа .NET була вибрана для проекту створення нового дефектоскопу через її високу масштабованість та продуктивність, що є критичними для обробки великих обсягів даних. Її багатомовність і крос-платформність також дозволяють

легко адаптувати програмне забезпечення до різних систем. Крім того, сильні можливості безпеки та управління пам'яттю на платформі .NET додатково підвищують надійність та ефективність додатків[15].

2.5.2 Мова програмування C#

Мова програмування C# – це об'єктно-орієнтована мова програмування, розроблена Microsoft як частина платформи .NET, офіційно представлена в 2000 році. Вона розроблялася як мова з простим, сучасним та загальним синтаксисом, призначена для створення широкого спектру додатків, від настільних до мобільних та веб-застосунків. C# поєднує принципи з мов програмування, як C++ та Java, і відома своєю сильною типізацією, автоматичним управлінням пам'яттю, та підтримкою концепцій, таких як асинхронне програмування та LINQ[16][17].

З розвитком C# вона постійно оновлювалася, включаючи нові функції та покращення у кожній версії. Це забезпечило високу продуктивність, надійність та безпеку додатків. Мова підтримує інтегроване середовище розробки (IDE) Visual Studio, що забезпечує потужні інструменти для розробки та відлагодження. Архітектура C# орієнтована на легкість розробки і має ряд важливих властивостей, таких як управління пам'яттю, об'єктно-орієнтоване програмування, і використання .NET Framework. Основні переваги мови програмування C# включають:

- C# підтримує об'єктно-орієнтоване програмування, що є ідеальним для великих, масштабованих проектів;
- завдяки системі збору сміття, розробники не потребують ручного керування пам'яттю, зменшуючи помилки та витоки пам'яті;
- C# розроблена для інтеграції з .NET, що надає доступ до великої бібліотеки коду та інструментів;

- C# має вбудовані механізми безпеки, які допомагають захистити код від вразливостей;
- З запуском .NET Core C# стала крос-платформною мовою, що розширює можливості її використання.

C# була обрана для проекту через ці переваги, а також завдяки її гнучкості, продуктивності та здатності впоратися з різноманітними вимогами проекту.

2.5.3 GnuPlot

Gnuplot – це програма для генерації дво- та тривимірних графіків, що була створена в 1986 році[18]. Вона підтримує багато типів графіків і може працювати на різних операційних системах. Gnuplot використовується для візуалізації наукових даних і підтримує різні формати виводу, включаючи інтерактивні веб-сторінки та графічні формати, такі як PNG, EPS, SVG, JPEG. Вона має текстовий інтерфейс користувача і використовується у багатьох галузях, включаючи інженерію, фізику, економіку та біологію. Gnuplot, з часом свого існування, значно розвинулася, отримавши багато оновлень та доповнень. Ця програма відома своєю здатністю обробляти різноманітні види даних і генерувати з них графіки. Її архітектура дозволяє працювати як з інтерактивними, так і зі скриптовими режимами, надаючи користувачам гнучкість у виборі способу візуалізації даних. Gnuplot широко використовується в академічних дослідженнях, інженерії та аналізі даних для створення якісних графіків та діаграм[19].

Gnuplot має широкий спектр можливостей для візуалізації даних:

- підтримка різних типів графіків: включає лінійні, стовпчасті, точкові, сплайн-графіки та багато інших;
- висока кастомізація: можливість детально налаштовувати стиль, кольори та шрифти графіків;

- підтримка багатьох вихідних форматів: Gnuplot може експортувати графіки у різні формати, включаючи PNG, EPS, SVG;
- скриптоване використання: підтримка автоматизації за допомогою скриптів для повторного використання та пакетної обробки;
- інтерактивність: можливість інтерактивної роботи з графіками, що полегшує аналіз даних.

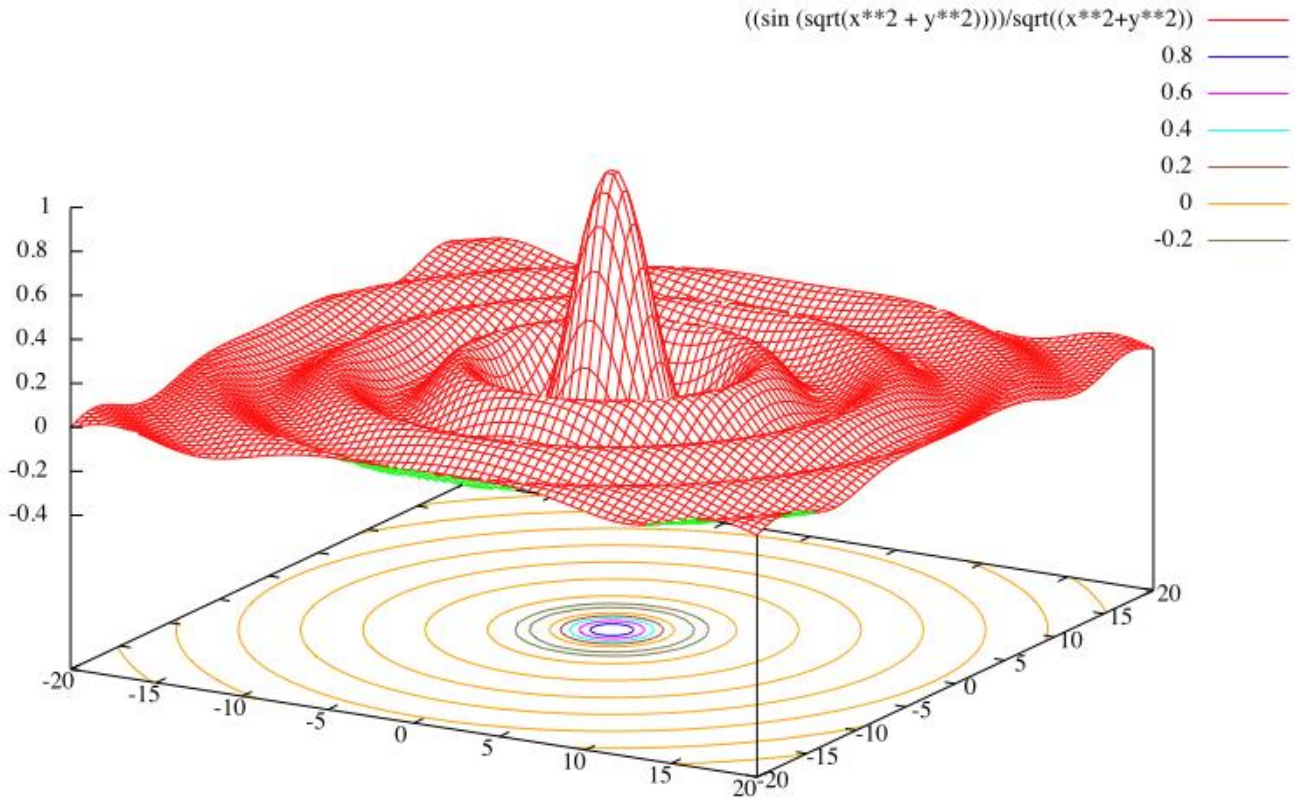


Рисунок 2.10 – Електромеханічний генератор вібрацій

Gnuplot була використана у проекті через її потужні можливості візуалізації даних, які є ключовими для аналізу та представлення результатів досліджень. Її висока кастомізація дозволяє точно налаштовувати графіки, що сприяє чіткому представленню складних даних. Скриптоване використання та підтримка багатьох форматів експорту також полегшує інтеграцію з іншими інструментами аналізу та документації[20]. Приклад можливостей GnuPlot можна переглянути на рисунку 2.10.

2.5.4 База даних

Файлова база даних – це система управління базами даних, в якій дані зберігаються у вигляді файлів на диску. Вона відрізняється від традиційних реляційних баз даних, оскільки не вимагає використання складних механізмів управління та структуризації даних, як наприклад, таблиць чи схем. Це робить файлові бази даних більш гнучкими та простішими для невеликих застосувань або проектів, де не потрібна висока складність обробки даних.

Переваги файлових баз даних включають:

- простота використання: легше створювати та управляти порівняно зі складними реляційними СУБД;
- легкість реалізації: можуть бути швидко реалізовані без великих витрат часу або ресурсів;
- гнучкість: можуть бути легко адаптовані під специфічні потреби проекту;
- відсутність залежності від складних систем: не потребують використання складних СУБД;
- ефективність для малих проектів: ідеальні для невеликих або менш складних застосувань.

Проте серед недоліків:

- обмежена масштабованість: вони не ідеальні для великих проектів або даних, що швидко зростають;
- відсутність ефективних механізмів обробки запитів: обробка даних може бути менш ефективною порівняно з реляційними СУБД;
- труднощі з одночасним доступом: вони не ефективно керують одночасним доступом до даних, що може викликати проблеми у багатокористувацьких системах;
- відсутність транзакційності: вони зазвичай не підтримують транзакції, що важливо для забезпечення цілісності даних;

- обмежена функціональність: вони не надають складних функцій обробки та аналізу даних, наявних у реляційних СУБД.

Файлова база даних підходить для прототипу проекту, оскільки вона проста в реалізації і не вимагає складного управління. Це робить її ідеальною для швидкого розвитку та тестування функцій. Її використання в проекті було обрано через легкість впровадження та низькі витрати часу на налаштування. Однак, для подальшого розвитку проекту та масштабування, може знадобитися перехід на більш потужну СУБД, яка зможе ефективніше обробляти збільшений обсяг даних та забезпечити кращу підтримку транзакцій та одночасного доступу.

Висновки до розділу 2

Існують численні архітектури, технології та методи для розробки систем і аналізу даних. Вибір відповідних технологій є критично важливим для успіху будь-якого проекту. Неправильний вибір може призвести до ускладнення розробки та надмірних витрат ресурсів. У цьому контексті був проведений глибокий аналіз існуючих архітектур, мов програмування, фреймворків, методів бізнес-аналізу, алгоритмів прогнозування та інших технічних засобів, що можуть бути використані для розв'язання поставленої задачі. У даному розділі детально розглядаються обрані засоби розробки та основні причини їх вибору. Зазначений набір технологій представляє собою ефективний інструментарій для розробки проекту, забезпечуючи необхідний баланс між функціональністю, продуктивністю та вартістю реалізації.

3 РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Загальна структура системи

На рисунку 3.1 зображена загальна архітектура (структура) системи:

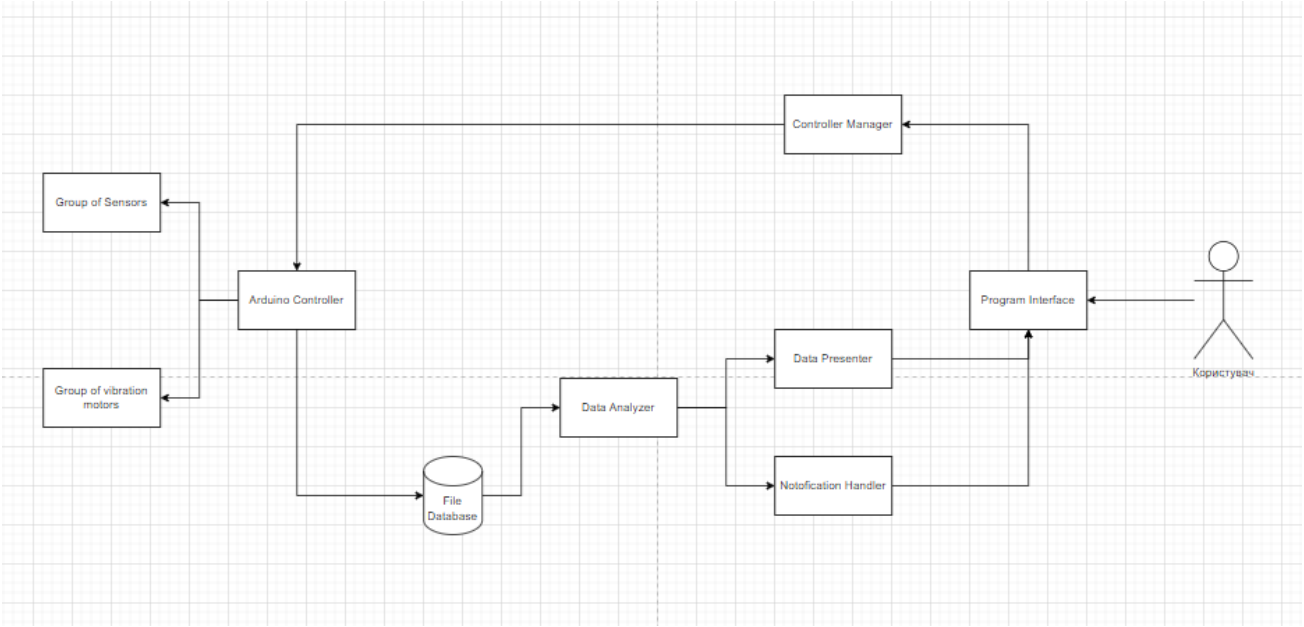


Рисунок 3.1 – Схема загальної структури системи

Схема, представляє загальну структуру системи дефектоскопа, розроблену для моніторингу стану інфраструктурних об'єктів. Система складається з кількох ключових компонентів та їх взаємодії, описаних нижче:

- група датчиків (Group of Sensors): ці датчики на основі п’єзоелементів відіграють важливу роль у виявленні змін реакції матеріалу на вібрації, на чому і заснована дефетометрія. Вони вимірюють параметри, критичні для оцінки стану інфраструктурних об'єктів. Дані, зібрані датчиками, відправляються до Data Analyzer для обробки;
- група генераторів вібрацій (Group of vibration motors): Використовується для створення вібраційних сигналів, які потім відстежуються групою сенсорів, Arduino Controller має змогу вмикати та вимикати генератори;

- Arduino controller: контролер апаратної частини, який здійснює контроль та обробку сигналів від датчиків також він керує роботою вібраційних моторів за заданими параметрами;
- file database: служить для довгострокового зберігання даних, що забезпечує можливість архівації, пошуку та аналізу історичних даних. база даних може містити великі обсяги інформації, яку система збирає протягом тривалого часу;
- data analyzer: аналізатор даних працює з інформацією, що зберігається в базі даних. Він використовує алгоритми для виявлення аномалій, тенденцій та шаблонів, які можуть свідчити про проблеми чи зміни у стані моніторингових об'єктів;
- data presenter: цей компонент відповідає за представлення даних користувачеві в зрозумілій і зручній формі. Це може бути дашборд, графіки, звіти тощо, що дозволяє користувачам легко інтерпретувати результати моніторингу;
- notification handler: обробник сповіщень генерує оповіщення для користувачів у разі виявлення критичних аномалій чи важливих змін у даних;
- controller manager: управляє всіма компонентами системи, включаючи arduino controller, забезпечуючи інтеграцію та синхронізацію роботи всієї системи;
- program interface: інтерфейс, через який користувач взаємодіє з системою, включаючи перегляд даних та управління налаштуваннями;
- користувач (user): остаточний елемент системи, який використовує всі перелічені вище компоненти для виконання моніторингу. Користувачі залежно від своїх ролей та потреб можуть переглядати інформацію, отримувати оповіщення та вживати необхідні заходи відповідно до отриманих даних.

За допомогою цієї системи користувачі можуть проводити комплексний моніторинг стану інфраструктурних об'єктів, швидко виявляючи потенційні дефекти чи зміни у стані об'єктів, що можуть вказувати на потребу в технічному обслуговуванні чи ремонті.

3.2 Компоненти апаратної частини

Розглянимо основні компоненти апаратної частини, серед яких: сенсори, генератори вібрацій та контролер Ардуіно.

3.2.1 Сенсори вібрацій

У системі моніторингу стану інфраструктури, яку було розроблено, використовуються сенсори на основі п'єзоелементів. Ці сенсори мають кілька переваг для виявлення вібраційних сигналів, що є ключовими для виявлення дефектів у металевих конструкціях. Ці датчики перетворюють вібраційні сигнали на електричний струм, що дозволяє точно вимірювати і реєструвати вібрації у конструкції.

Основна мета застосування цих сенсорів полягає у забезпеченні безперервного та точного моніторингу стану інфраструктурних об'єктів. П'єзоелектричні датчики вібрації фіксують найменші зміни у вібраціях, що дозволяє своєчасно виявити аномалії, які можуть свідчити про початкові стадії виникнення дефектів[21].

Для максимальної ефективності сенсори розташовуються у критичних точках конструкції, де найбільш ймовірно виникнення дефектів або де вони можуть бути найкраще зафіксовані. Розміщення сенсорів проектується з урахуванням геометрії, матеріалів та типів навантажень, яким піддається конструкція, також сенсори можна розташовувати рівномірно покриваючи увесь об’єкт інфраструктури.

Для детального уявлення про розміщення та інтеграцію п'єзоелектричних сенсорів у структуру моніторингу, дивіться рисунок 3.2, який ілюструє їх розташування на конструкції. Рисунок надає візуальне представлення конфігурації сенсорів та їх з'єднання з центральним контролером для обробки та аналізу даних.

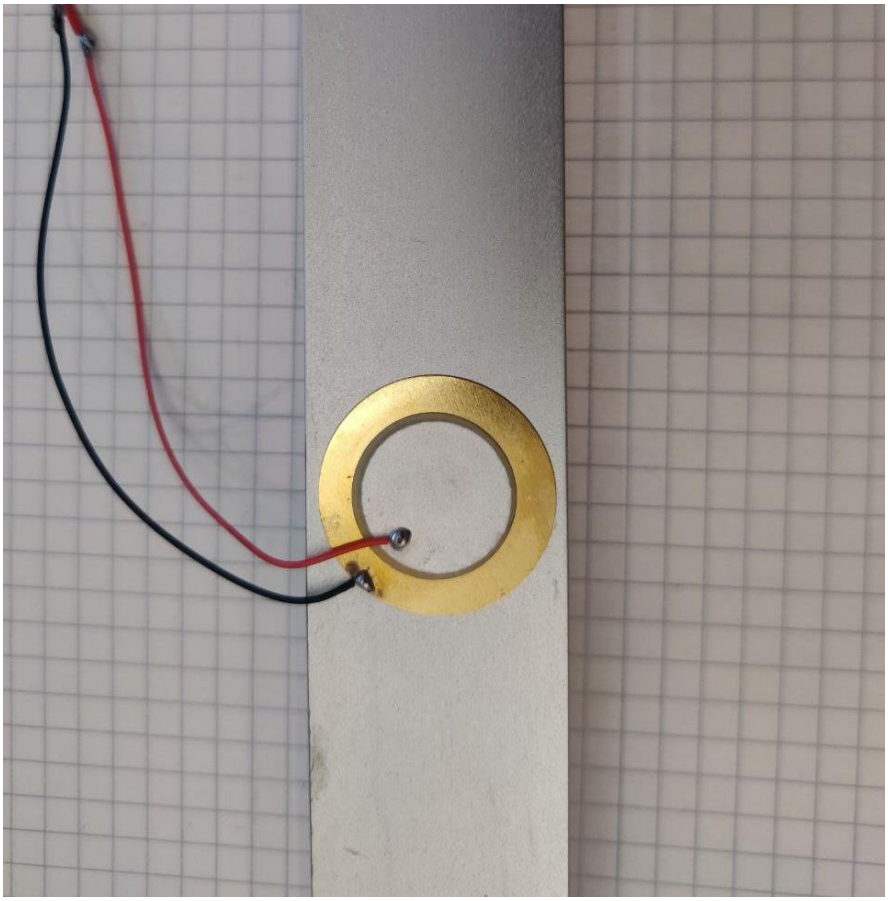


Рисунок 3.2 – Приклад розміщення сенсора

На цьому зображенні ви можете бачити п'єзоелектричний датчик вібрації, який прикріплено до металевої конструкції. Датчик складається з круглої п'єзоелектричної пластини, до якої підключено два проводи (червоний і чорний), що забезпечують електричне з'єднання з обладнанням для реєстрації та аналізу вібраційних сигналів.

П'єзоелектричні датчики вібрації ефективні для виявлення та вимірювання вібрацій у різних частинах інфраструктурних об'єктів. Вони використовують

властивості п'єзоелектричних матеріалів, які генерують електричний заряд у відповідь на механічний тиск, тобто вібрації. Цей процес перетворення дозволяє точно фіксувати вібраційну активність, яка може вказувати на наявність дефектів або зміни в інтегральному стані матеріалу.

Зображення також демонструє, що датчик розміщено на поверхні, яка дозволяє максимально точно передавати вібраційні хвилі від конструкції до датчика. Проводи можуть бути підключені до мікроконтролера або іншої системи збору даних для подальшого аналізу отриманих сигналів.

3.2.2 Генератори вібрацій

У цьому розділі ми розглянемо ключові аспекти вібромоторів, які використовуються як генератори вібрацій у системах моніторингу стану інфраструктурних об'єктів.

Типи вібромоторів:

- eccentric rotating mass (ERM): це традиційні вібромотори, в яких вібрація створюється за допомогою обертання ексцентрикового масиву;
- linear resonant actuator (LRA): вони виробляють вібрацію шляхом швидкого поступального руху ваги вперед і назад по одній лінії.

Для своїх потреб я використав, тип, в яких вібрація створюється за допомогою обертання ексцентрикового масиву, через його наявність в магазині, але це не впливає на роботу пристрою, головне, щоб вібрації були однаковими впродовж використання пристрою для моніторингу, бо при зміні типу, система розпізнає це як виникнення аномалії. ERM вібромотори створюють вібрацію за допомогою обертання несиметричного вантажу, прикріпленого до вала двигуна. Швидкість та амплітуда вібрації можуть бути змінені шляхом регулювання швидкості обертання вантажу[23].

Вібромотори використовуються для створення контрольованих вібрацій, які у свою чергу повинні бути записані через сенсори, те як матеріал реагує на вібрації і показує наявність дефектів, а якщо більш детально то при зміні характеристики коливань, можна припустити і зміну у структурі матеріалу, тобто виникненню дефекту, якщо брати початковий стан за ідеальний.

Вібромотори розташовуються на конструкції, зазвичай в місцях, які рівновіддалені від сенсорів, для створення рівномірного розподілу вібраційних хвиль по всій структурі. Це забезпечує, що вібрації можуть бути зафіксовані сенсорами з різних локацій для всебічної оцінки стану конструкції.

Кожен вібромотор може обслуговувати один або кілька сенсорів залежно від їх розміщення та потреб системи. Це дозволяє оптимізувати кількість вібромоторів, необхідних для ефективного моніторингу великих конструкцій. Для ілюстрації як вібромотори розташовуються відносно п'єзоелектричних сенсорів та інших елементів конструкції, дивіться рисунок 3.3. Рисунок надає інформацію про фізичне розміщення вібромоторів, їх з'єднання з електричною мережею та їх взаємодію з системою сенсорів для створення точного та контрольованого вібраційного навантаження.

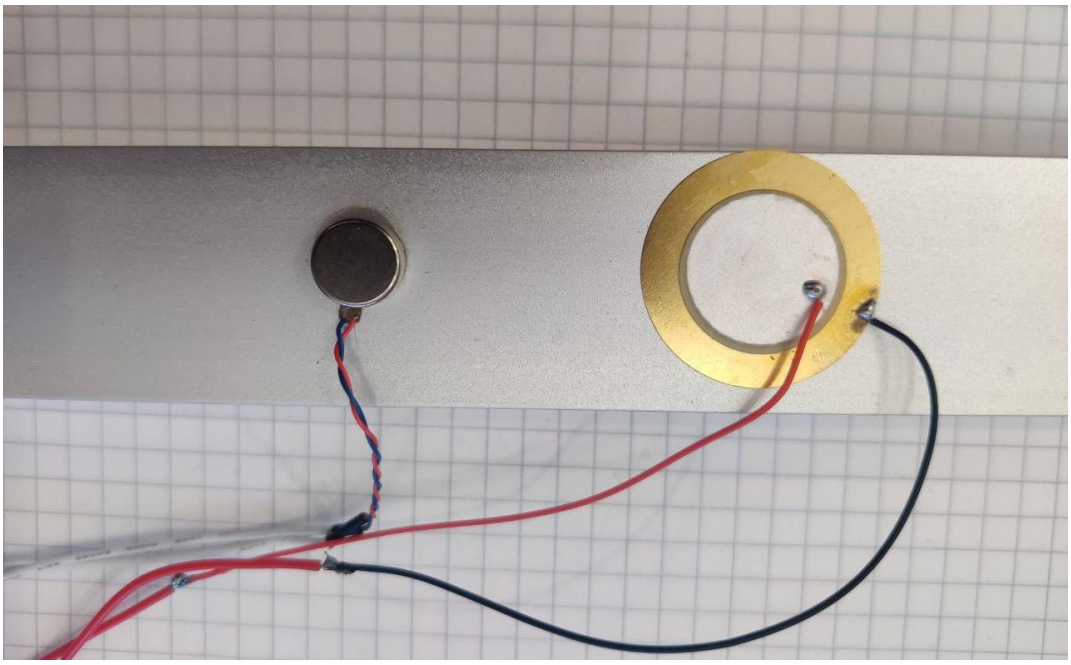


Рисунок 3.3 – Приклад розміщення вібромотора

На фотографії ви можете бачити вібромотор, який розташований поруч з п'єзоелектричним датчиком вібрації. Вібромотор – це круглий компонент з металевої оболонки, який, зазвичай, містить в собі ротор з ексцентриковим вантажем, що при обертанні створює вібрацію.

Цей мотор призначений для генерації вібраційних сигналів, які потім можуть бути виявлені п'єзоелектричним датчиком. Вібромотор підключений до електронної системи за допомогою проводів (часто червоного та синього кольорів, які вказують на полярність). Вібрації, створені вібромотором, передаються на конструкцію, на якій закріплені ці компоненти, що дозволяє симулювати реальні умови експлуатації та тестувати чутливість сенсора.

Розміщення вібромотора відносно сенсора є ключовим для ефективної роботи системи. Зазвичай вібромотори розташовують таким чином, щоб забезпечити максимальний розподіл вібраційних хвиль та одночасно уникнути прямого впливу на сенсори, що могло б викликати перевантаження або пошкодження сенсорів. Це також забезпечує, що сенсори можуть вимірювати вібрації з різних точок, що покращує загальну картину стану конструкції.

Вібромотор може обслуговувати один або кілька сенсорів залежно від їх розміщення та налаштування системи. Це означає, що один мотор може бути достатнім для активації декількох сенсорів в різних точках конструкції, дозволяючи цим чином оптимізувати кількість необхідного обладнання.

Для наглядності, на прилеглому рисунку можна бачити як вібромотор і п'єзоелектричний датчик розташовані на конструкції. Ця візуалізація допомагає зрозуміти взаємодію між вібромотором та сенсором та є важливою для розуміння загальної роботи системи моніторингу.

3.2.3 Контролер Arduino

Контролер Arduino є центральним вузлом у системі моніторингу стану інфраструктури. Arduino – це відкрита платформа на основі простої плати

мікроконтролерів та розробницького середовища, що дозволяє виконувати комплексні завдання обробки даних та управління периферійними пристроями. Контролер Arduino виконує кілька критичних функцій у системі:

- збір даних: Arduino зчитує сигнали від п'єзоелектричних сенсорів та інших датчиків системи;
- обробка даних: виконує первинну обробку даних, таких як фільтрація, підсилення сигналу, а також перетворення з аналогової форми у цифрову;
- управління вібромоторами: Ардуіно може контролювати роботу вібромоторів, регулюючи їхню частоту та амплітуду вібрацій;
- комунікація: надсилає оброблені дані на комп'ютер чи інші пристрої для подальшого аналізу;
- інтерфейс користувача: може надавати інформацію користувачу через різноманітні інтерфейси, такі як світлодіоди, дисплеї або безпосередньо через середовище розробки.

Arduino використовує програмований мікроконтролер, який може бути запрограмований для виконання широкого спектру завдань. Він включає в себе аналого-цифрові конвертори для зчитування аналогових сигналів від сенсорів, цифрові входи/виходи для керування зовнішніми пристроями, та серійні порти для комунікації. Програмне забезпечення для Arduino пишеться на мові, яка схожа на C/C++, що забезпечує гнучкість та потужність для реалізації необхідної логіки обробки та управління. Рисунок 3.4 демонструє, як контролер Arduino інтегровано у систему моніторингу. Він показує зв'язки між Arduino та іншими компонентами системи, включаючи сенсори, вібромотори, та інші периферійні пристрої. На рисунку також видно, як дані пересилаються до та з контролера, що ілюструє його роль як вузлового пункту в обробці та передачі інформації.

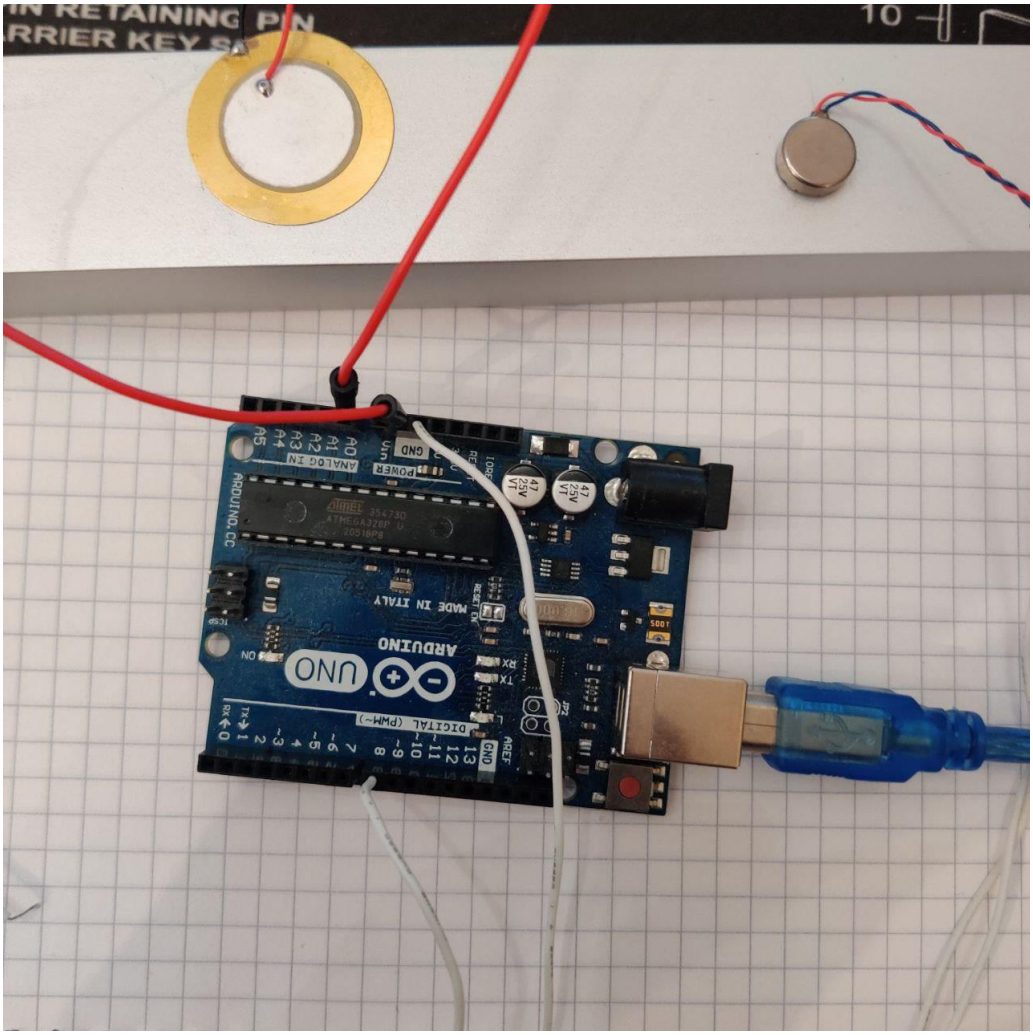


Рисунок 3.4 – Контролер Arduino

На цьому зображенні зображено комплект обладнання, що використовується в системі моніторингу стану конструкції.

Контролер Arduino Uno: Центральний елемент – плата Arduino Uno, яка є однією з найпопулярніших моделей Arduino завдяки своїй гнучкості та простоті використання. Вона оснащена мікроконтролером ATmega328P і містить низку входів/виходів для підключення різних сенсорів та актуаторів. Плата з'єднана з комп'ютером через USB кабель, який забезпечує живлення та передачу даних.

П'єзоелектричний сенсор вібрації: Над платою Arduino знаходиться круглий п'єзоелектричний датчик, прикріплений до тестової конструкції для моніторингу. Цей сенсор перетворює механічні вібрації на електричний сигнал, який потім може бути оброблений контролером Arduino.

На зображенні показано, як компоненти системи можуть бути фізично інтегровані. Arduino Uno слугує як мозок системи, збираючи та обробляючи дані від п'єзоелектричного сенсора, а також контролюючи роботу вібромотора. Взаємодія між цими компонентами є фундаментальною для забезпечення ефективного моніторингу стану конструкції.

Використання цих трьох компонентів разом дозволяє створювати систему, здатну не тільки виявляти існуючі дефекти через аналіз вібраційних сигналів, але й допомагає в проведенні профілактичних оглядів та тестуванні матеріалів на міцність. Розглянемо електричну блок-схему пристрою рисунок 3.5:

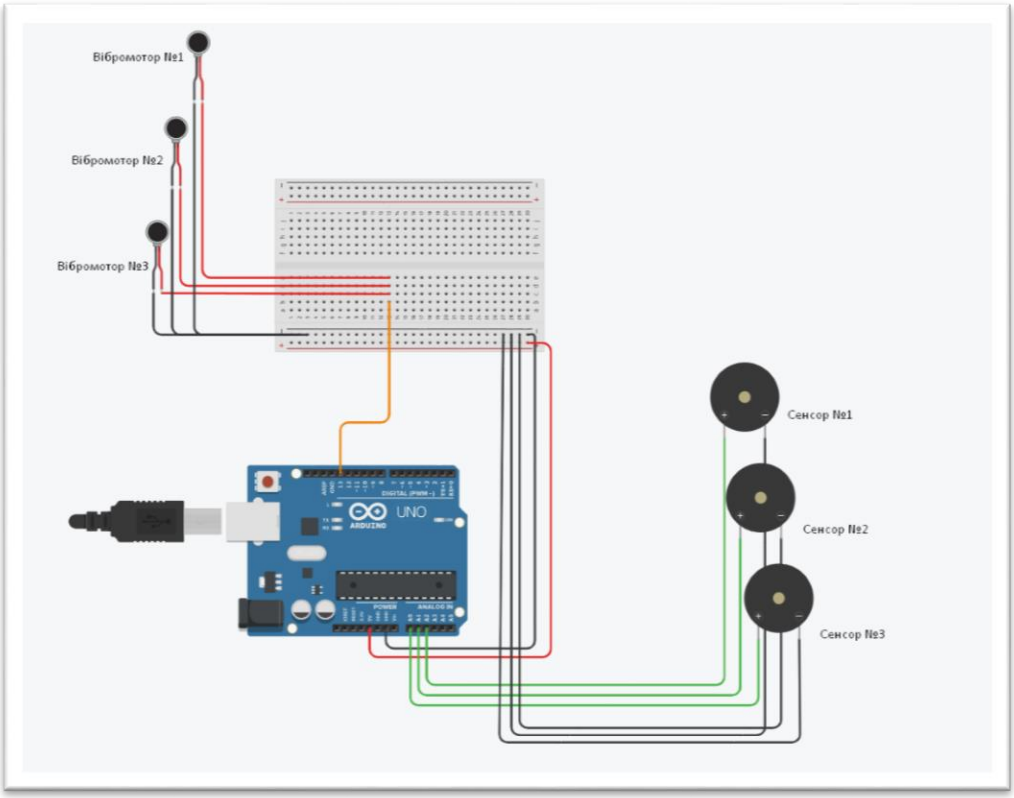


Рисунок 3.5 – Електрична блок-схема пристрою

На зображенні електричної блок-схеми ми бачимо підключення компонентів системи до плати Arduino Uno. Ця плата використовується як центральний контролер для взаємодії між сенсорами, вібромоторами та іншими

компонентами системи. Її основні елементи включають цифрові та аналогові входи/виходи, порти живлення, заземлення та порти комунікації.

На схемі позначено три вібромотори, які підключені до цифрових виходів Arduino Uno через макетну плату (breadboard). Кожен вібромотор керується окремим виходом, що дозволяє індивідуально контролювати роботу кожного мотора, але за програмою вони працюють одночасно. Червоні проводи, забезпечують живлення, а чорні проводи — землю.

П'єзоелектричні сенсори підключені до аналогових входів на Arduino. Сенсори перетворюють вібрації на аналоговий електричний сигнал, який потім може бути прочитаний через аналогові піни. Зелені проводи від сенсорів представляти сигнальні лінії, які передають аналогові сигнали до контролера.

USB кабель підключений до Arduino Uno для програмування, живлення та серійної комунікації з комп'ютером. Біла макетна дошка використовується для тимчасового з'єднання компонентів без пайки, що дозволяє легко модифікувати схему.

3.3 Компоненти програмної частини

У цьому розділі розглянемо компоненти програмної частини, до якої відносяться файлове сховище, аналізатор даних, візуалізатор даних, контролер сповіщень, контролер ардуіно, програмний інтерфейс.

3.3.1 База даних

База даних у цьому контексті виконує функцію централізованого сховища для запису та зберігання результатів тестів, отриманих від системи моніторингу стану конструкції, що управляється за допомогою контролера Arduino.

Замість традиційної реляційної бази даних, для цього проекту використовується файлове сховище, що забезпечує простоту та зручність зберігання даних без необхідності налаштування складних баз даних.

Результати кожного тесту записуються у окремий файл, що полегшує доступ до інформації про певний тест та дозволяє легко аналізувати дані за певні періоди.

Кожен запис у файлі має формат <час>: <показник сенсору 1>,<показник сенсору 2>, ..., <показник сенсору N>, де N – кількість сенсорів у системі. Часова мітка <час> фіксує точний момент зняття показів, забезпечуючи можливість відстежувати динаміку зміни показників. Показники сенсорів зберігаються у форматі, що дозволяє легко відокремити дані кожного сенсора для подальшого аналізу.

Система може бути запрограмована таким чином, що після завершення кожного тесту або з заданою періодичністю, дані автоматично зберігаються у файлове сховище. Файли можуть бути відсортовані або організовані за датою та часом, для зручності пошуку та витягу даних. Хоча використання файлового сховища є оптимальним для малих або середніх проектів, систему можна модернізувати для використання більш складних рішень управління базами даних, якщо з'явиться потреба обробляти більші обсяги даних або забезпечити більш складний аналіз.

Для наглядної демонстрації архітектури файлового сховища та процесу збору даних, можна переглянути рисунок 3.6 та 3.7 які ілюструють структуру файлів, методи запису та організацію даних у системі.

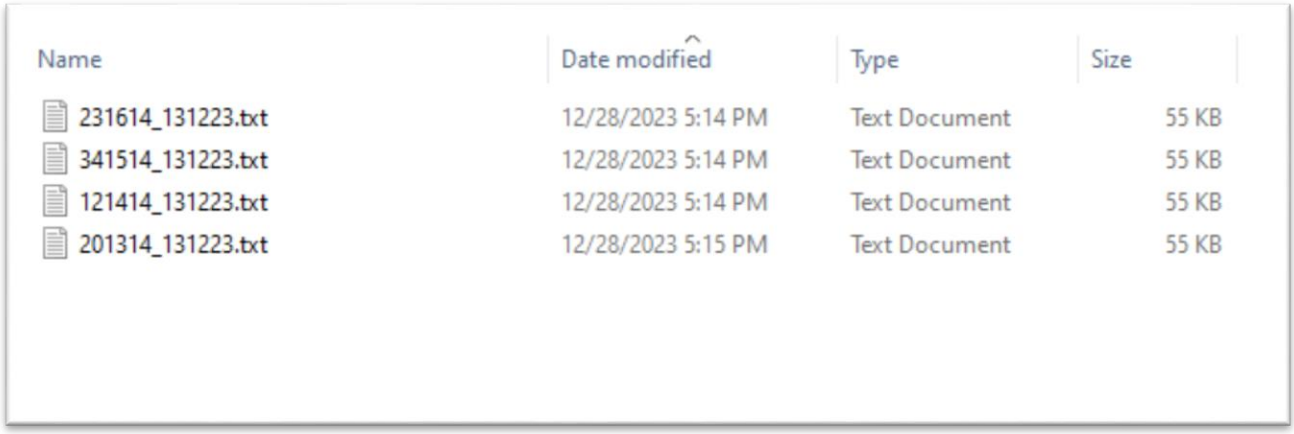


Рисунок 3.6 – Приклад збереження файлів

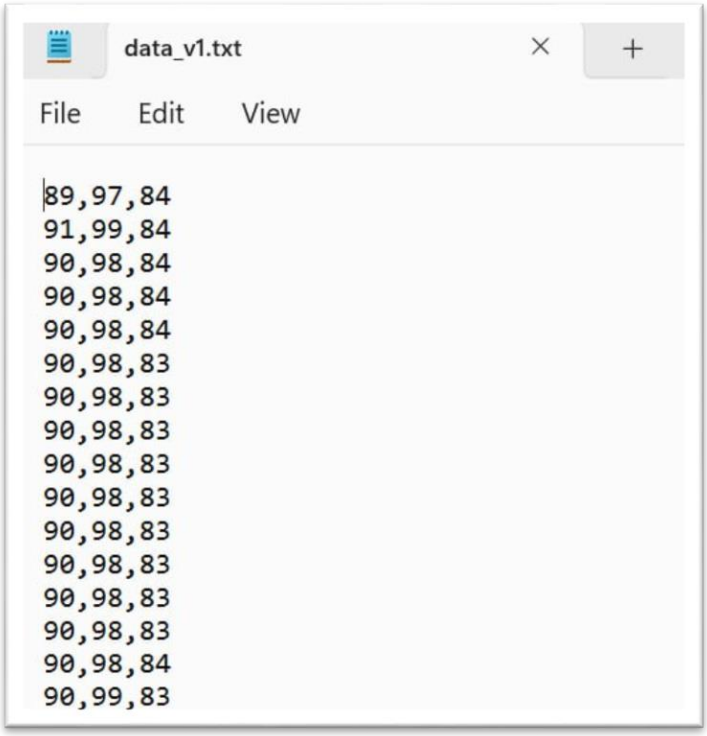


Рисунок 3.7 – Приклад збережених даних у файлі для 3-х сенсорів

Отримані файли можуть бути імпортовані в програми для аналізу даних, такі як Excel, MATLAB, Python або спеціалізоване програмне забезпечення для візуалізації та аналізу.

3.3.2 Аналізатор даних

Аналізатор даних — це модуль програмного забезпечення системи моніторингу, який забезпечує обробку та аналітичний аналіз даних, отриманих від контролера Arduino. Аналізатор даних імпортує файл з новими записами від Arduino, коли такий стає доступним. Програма порівнює нові дані з історичними даними, отриманими з того ж об'єкту інфраструктури, для виявлення аномалій або змін у поведінці системи.

Аналізатор обчислює основні статистичні показники, такі як мінімальні, максимальні та середні значення вібрацій для кожного сенсору. Використовує методи обчислення відхилень від середнього значення або відхилень від встановленого порога, для ідентифікації нехарактерних вібрацій.

Алгоритми виявляють тренди у даних, такі як постійне збільшення амплітуди вібрацій, що може вказувати на поступове погіршення стану об'єкту. Встановлення взаємозв'язків між показниками різних сенсорів для виявлення складних взаємодій у системі.

Після завершення аналізу оброблені дані передаються до модуля візуалізації, де вони можуть бути відображені у формі графіків, теплових карт, дашбордів тощо, для інтерпретації інженерами або системами автоматизованого рішення.

Цей модуль є ключовим для підтримки прийняття рішень щодо технічного обслуговування або ремонту, забезпечуючи високий рівень надійності і безпеки об'єктів інфраструктури.

Модуль оснащений консольним інтерфейсом для взаємодії з користувачем. Це дозволяє отримувати результати у вигляді текстових повідомлень у консолі. Після завершення обчислень, модуль автоматично записує результати в окремий файл. Цей файл включає в себе назву вхідного файлу з даними та результати їх обчислення, що забезпечує чіткий аудит та звітність аналітичних процесів, а також використовується при майбутніх тестах.

Формат вихідного файлу з результатами такий: DataFileName: AnalysisResults, де DataFileName є ім'ям файлу з даними, а AnalysisResults містить обчислені показники та висновки.

3.3.3 Візуалізатор даних

Модуль візуалізації даних є важливим компонентом у системі моніторингу стану інфраструктури, оскільки він дозволяє операторам швидко та інтуїтивно зрозуміти зібрану інформацію. Використання графічного зображення даних може значно підвищити ефективність аналізу та діагностики.

Модуль візуалізації активується безпосередньо з модуля аналізу даних, який надсилає оброблені дані для відображення. Після завершення обчислень модулем аналізу, візуалізаційний модуль автоматично генерує візуальне представлення отриманих результатів.

Модуль використовує пакет GnuPlot, адаптований для C#, який є потужним інструментом для створення наукових графіків. GnuPlot дозволяє генерувати графіки в різних форматах, включаючи інтерактивні графіки, які можна переглядати в окремих вікнах, або як статичні зображення для включення в звіти. GnuPlot підтримує широкий спектр форматів виведення, включаючи векторні та растрові формати зображень, що робить його ідеальним для використання у наукових публікаціях, технічних звітах та презентаціях. Інтерактивні можливості GnuPlot, такі як масштабування, обертання та зміна перспективи графіків, роблять його корисним інструментом для дослідження та аналізу даних.

Особливістю GnuPlot є його здатність до детальної налаштування графіків, включаючи стилі ліній, маркерів, шрифтів та кольорів, що дає користувачам величезні можливості для створення візуально привабливих і інформативних графіків. Його використання в модулі візуалізації даних для системи моніторингу

стану інфраструктури підкреслює його здатність ефективно представляти складні набори даних у зрозумілій і доступній формі.

На одному графіку відображаються показники, які були зареєстровані сенсорами під час поточного тесту, а також показники попереднього тесту, це можна побачити на рисунку 3.8. Ця порівняльна візуалізація допомагає оператору виявити будь-які значущі відхилення або тренди, що можуть вказувати на зміни в стані об'єкту.

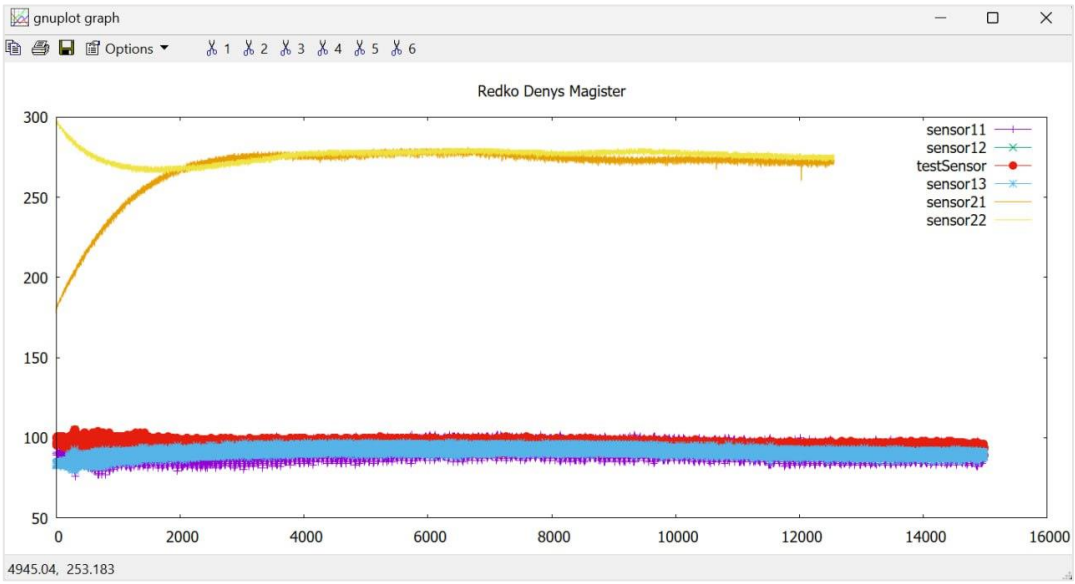


Рисунок 3.8 – Приклад графіку

У центрі є графік з різнокольоровими лініями, які представляють дані з різних датчиків, позначених як sensor11, sensor12, sensor13, sensor21, sensor22. Лінії розташовані на координатній сітці з горизонтальною віссю, яка, показує індекс зразка (значення від 0 до більше ніж 16000), та вертикальною віссю, що показує значення датчиків (від 0 до 300).

Жовта лінія (sensor11) показує значне збільшення на початку, після чого вона стабілізується на високому рівні. Це вказує на зміни у стані об’єкта на початку бо існує проміжок часу коли вібромотори не набрали максимальну потужність. Інші лінії (sensor12, sensor13) знаходяться на нижчому і відносно стабільному рівні, різниця між показниками sensor11, sensor12, sensor13 та,

sensor21, sensor22 вказує на зміну у структурі матеріалу, у даному випадку була змінена поверхня на якій розташовувася металевий профіль.

Додатково до графічного відображення, модуль також виводить у консоль результати поточного тесту та десяти попередніх тестів, приклад яких можна побачити на рисунку 3.9. Консольний вивід може включати статистичні дані та інші обчислені показники, що спрощує порівняння поточних даних з історичними.

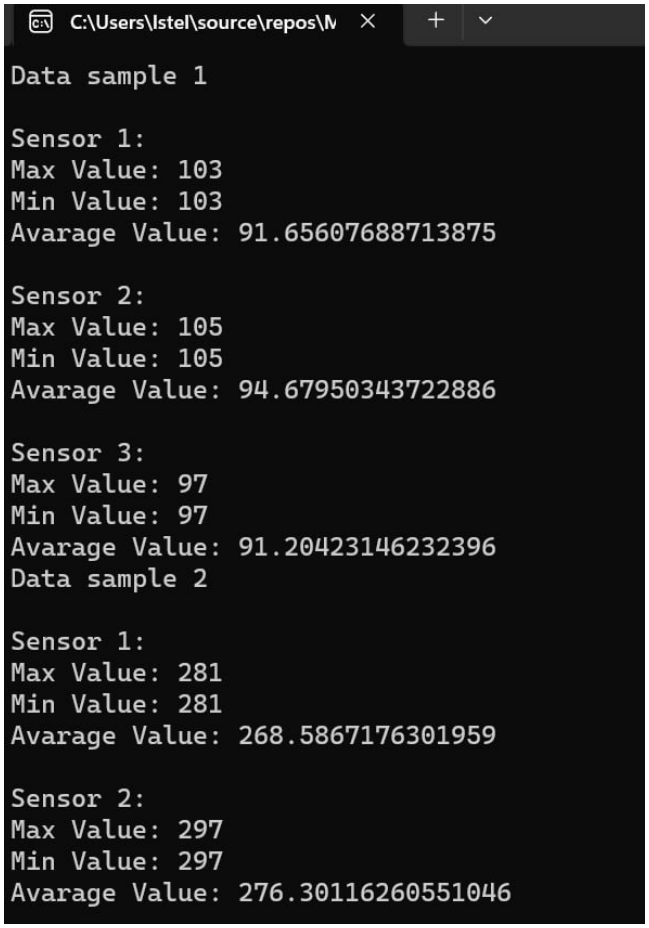


Рисунок 3.9 – Таблиця для порівнянь обчислених значень

Модуль візуалізації може бути використаний для презентації даних у реальному часі під час моніторингу, а також для генерації звітів, що можуть бути використані для технічних обговорень або в документації.

Завдяки інтеграції з модулем аналізу, автоматизованому виведенню графіків та консольного виводу, модуль візуалізації даних відіграє ключову роль

у забезпеченні швидкого та ефективного розуміння стану об'єктів інфраструктури та прийняття своєчасних рішень.

3.3.4 Контролер сповіщень

Модуль контролю сповіщень є важливою частиною автоматизованої системи моніторингу інфраструктури, оскільки він дозволяє операторам бути в курсі критичних змін без необхідності постійного спостереження за системою. Цей модуль забезпечує цілодобовий нагляд за об'єктами інфраструктури і автоматично надсилає сповіщення при виявленні значущих подій або тенденцій.

Оператор може встановити певні порогові значення для показників від сенсорів, при досягненні або перевищенні яких система активує сповіщення.

Ці параметри можуть включати максимальні та мінімальні значення, статистичні відхилення, аномальні частоти вібрацій, або специфічні для додатку умови. Модуль контролю сповіщень використовує алгоритми для аналізу даних та визначення, коли актуальні показники виходять за рамки встановлених критеріїв. При виявленні потенційної проблеми система автоматично ініціює процедуру сповіщення.

Сповіщення можуть бути відправлені через різні канали, включаючи електронну пошту (EMail), SMS на телефон, або інші методи, що підтримуються оператором.

Для цього модулю було використано сервіс AWS SNS. Використання сервісу AWS SNS (Amazon Web Services Simple Notification Service) дозволяє легко інтегрувати різноманітні канали сповіщень та забезпечити швидку та надійну доставку повідомлень. AWS SNS є масштабованим, гнучким і високонадійним сервісом, який дозволяє швидко налаштовувати та розсилати сповіщення. Сервіс дозволяє керувати великою кількістю сповіщень і забезпечує автоматизацію процесів з мінімальною затримкою між виявленням події та доставкою повідомлення. Завдяки інтеграції з надійним сервісом AWS SNS,

модуль контролю сповіщень забезпечує важливий рівень зв'язку у системі моніторингу, підвищуючи її надійність та реактивність, діаграму роботи AWS SNS можна побачити на рисунку 3.10.

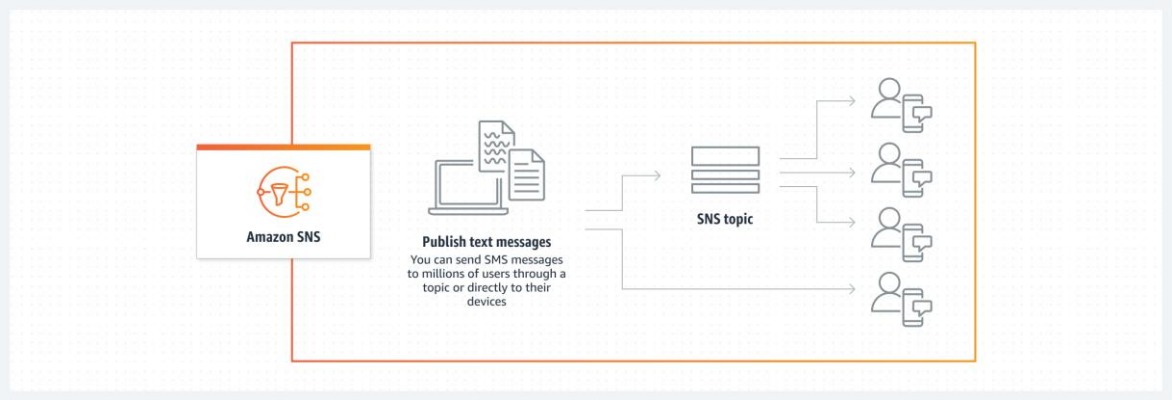


Рисунок 3.10 – Діаграма роботи AWS SNS

Amazon Web Services Simple Notification Service (AWS SNS) є хмарним рішенням, яке дозволяє розсилати сповіщення або повідомлення в реальному часі до великої кількості отримувачів. SNS надає можливість використовувати різноманітні канали сповіщень, такі як SMS, електронна пошта, Amazon SQS черги та AWS Lambda функції[24].

Сервіс працює на принципі «публікуй-підписуйся», де ви створюєте тему, або «topic», до якої можуть підписуватися різні служби або особи. Кожен раз, коли до теми публікується нове повідомлення, всі підписані отримувачі автоматично отримують це повідомлення через обраний ними спосіб отримання.

Автоматизовані сповіщення дозволяють зменшити кількість часу, який оператори витрачають на моніторинг даних, фокусуючись лише на важливих або критичних змінах. Це забезпечує ефективне використання часу оператора та дозволяє швидко реагувати на потенційні проблеми.

Для цього проекту було використана бібліотека яка спрощує надсилання повідомлень, розроблена для .Net, вона спрощує конфігурацію сервіса, авторизацію, інтеграцію з AWS SNS, дозволяючи розробникам швидко налаштувати сповіщення і розсилки без глибокого занурення в технічні деталі

сервісу. Завдяки цій бібліотеці, весь процес роботи з SNS стає більш інтуїтивним та ефективним, оскільки значною мірою усувається необхідність ручного налаштування та кодування низькорівневих запитів. Розробники можуть використовувати зрозумілі абстракції та методи високого рівня для управління темами, підписками та повідомленнями, зосереджуючись на бізнес-логіці своєї програми, а не на інфраструктурних задачах. Приклад налаштування можна побачити на рисунку 3.11.

```
public class SensorDataProcessor
{
    private double MaxLimit;
    private double MinLimit;
    private AmazonSimpleNotificationServiceClient snsClient;

    public SensorDataProcessor(ILimitsProvider _limitProvider)
    {
        snsClient = new AmazonSimpleNotificationServiceClient();
        MaxLimit = _limitProvider.Max;
        MinLimit = _limitProvider.Min;
    }

    public async Task CheckAndNotifyAsync(IEnumerable<SensorData> sensorDataList
    {
        foreach (var data in sensorDataList)
        {
            if (data.MaxValue > MaxLimit || data.MinValue < MinLimit)
            {
                await NotifyAsync($"Виявлено відхилення: Max: {data.MaxValue}, M
            }
        }
    }

    private async Task NotifyAsync(string message)
    {
        var request = new PublishRequest
        {
            TopicArn = TopicArn,
            Message = message
        };

        await snsClient.PublishAsync(request);
    }
}
```

Рисунок 3.11 – Виклик сповіщення

Використання AWS SNS в .NET реалізується через AWS SDK для .NET, який надає розробникам набір інструментів для легкої інтеграції з сервісами Amazon Web Services. Це SDK дозволяє розробникам використовувати C# або інші .NET-сумісні мови для взаємодії з SNS, включаючи створення тем,

публікацію повідомлень та управління підписками, що відкриває можливості для ефективної автоматизації сповіщень у індустріальних системах.

3.3.5 Контролер ардуіно

Модуль контролера Arduino є ключовою частиною системи, що забезпечує інтеграцію програмної та апаратної частин. Цей модуль відіграє роль посередника, який передає команди з комп'ютера на плату Arduino та збирає дані з апаратної частини для подальшого аналізу та обробки на комп'ютері.

У цьому контексті, програмування на Arduino виконується за допомогою спеціалізованого IDE, яке пропонує зручні інструменти для написання, тестування та завантаження коду на мікроконтролер. Завдяки своїй відкритості та гнучкості, Arduino дозволяє швидко адаптувати апаратну частину до нових вимог проекту, що є важливим у випадках, коли потрібно експериментувати або впроваджувати нові технологічні рішення. Крім того, широкий спектр доступних бібліотек та підтримка спільноти розробників роблять Arduino надзвичайно популярним вибором для прототипування та розробки в умовах обмежених бюджетів, зберігаючи при цьому високі стандарти якості та надійності.

Модуль може надсилати команди з комп'ютера до Arduino, які можуть включати запуск або зупинку збору даних, налаштування параметрів сенсорів, управління вібромоторами, або ініціацію діагностичних тестів. Arduino збирає дані з підключених сенсорів та інших пристроїв, обробляє ці дані (якщо це необхідно) та передає їх назад до комп'ютера через цей модуль, приклад конфігурації комунікації через серійний порт можна побачити на рисунку 3.12.

```
public static System.IO.Ports.SerialPort serialPort1;

    private void establishConnection()
    {
        serialPort1 = new System.IO.Ports.SerialPort("COM4");
        serialPort1.BaudRate = 115200;

        serialPort1.Open();
    }

    static void Main(string[] args)
    {
        Program p = new Program();
        p.establishConnection();

        string i;
        while (true)
        {
            Console.Write("Enter command:  ");
            i = Console.ReadLine();
            if (i == "exit")
            {
                serialPort1.Close();
                break;
            }
            else if (i == "Start Test")
            {
                serialPort1.Write("Start Test\n");
            }
        }
    }
```

Рисунок 3.12 – Конфігурація комунікації через серійний порт у C#

Принцип роботи модуля:

- ініціація команди: команда генерується на комп'ютері, наприклад, через графічний інтерфейс користувача або консольну команду;
- передача команди: команда пересилається до Arduino через серійний зв'язок по USB кабелю;
- виконання команди: Arduino отримує команду, обробляє її та виконує необхідні дії, як-от збір даних або управління периферійними пристроями;
- зворотний зв'язок: Arduino надсилає статус виконання команд і зібрані дані назад до комп'ютера для подальшої обробки.

Використовуючи серійний порт для комунікації, програма на C# відправляє дані до Arduino через USB з'єднання. Програма на C# використовує функції

SerialPort.Write для відправки команд до Arduino, які Arduino інтерпретує та виконує, приклад коду ардуіно який реєструє команду StartTest і привєзує її до методу, який починає процес тестування, приклад можна побачити на рисунку 3.13, де можна побачити приклад налаштування серійного порту з швидкістю передачі даних у 115200.

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

  sCmd.addCommand("startTest",  startTest);
```

Рисунок 3.13 – Додавання команди StartTest до серійного інтерфейсу

Цей модуль є фундаментальним для забезпечення ефективної взаємодії між програмними та апаратними компонентами системи, дозволяючи операторам керувати процесами збору даних, моніторингу та управління у реальному часі.

Висновки до розділу 3

Реалізація системи моніторингу інфраструктури, яка була розглянута, є комплексним рішенням, яке включає в себе інтеграцію апаратних сенсорів і контролерів з програмним забезпеченням. Використання контролерів Arduino як центрального вузла для збору даних від сенсорів і управління актуаторами, такими як вібромотори, дозволяє забезпечити точність і оперативність у виявленні аномалій. Файлова система для збереження даних тестів відіграє роль надійного сховища для історичних даних та результатів аналізу, що сприяє легкому доступу та відстеженню трендів з часом. Модуль аналізу даних, розроблений на мові C#, забезпечує глибоке вивчення зібраних даних, використовуючи різноманітні алгоритми для оцінки параметрів, таких як максимальні, мінімальні та середні значення. Інтеграція з GnuPlot через адаптований C# пакет надає потужні інструменти для візуалізації даних,

дозволяючи операторам системи ефективно порівнювати поточні показники з історичними даними.

Важливим аспектом системи є модуль контролю сповіщень, який, за допомогою сервісу AWS SNS, дозволяє операторам отримувати своєчасні оповіщення про критичні події або значні зміни у показниках. Це мінімізує ризики, пов'язані з потенційними пошкодженнями інфраструктури та забезпечує можливість швидкого реагування на несправності.

4 ІНСТРУКЦІЯ КОРИСТУВАЧА

4.1 Огляд можливостей системи для аналізу

Покроково розглянемо всю функціональність, що реалізована в системі. Інтерфейс програми представлен у вигляді консолі. При запуску програми ми бачимо консоль, в якій зображені підказки керування, що можна побачити на рисунку 4.1.

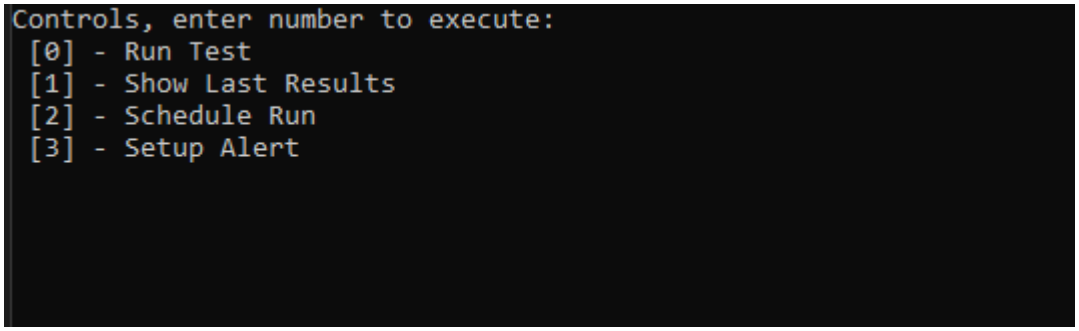


Рисунок 4.1 – Головна консоль застосунку

На малюнку зображено консольний інтерфейс програми, який містить меню з чотирма опціями для користувача. Кожна опція позначена номером та коротким описом дії, яку вона виконує:

- опція [0] – «Run Test»: ініціює процес тестування;
- опція [1] – «Show Last Results»: відображає результати тестів;
- опція [2] – «Schedule Run»: дає можливість запланувати автоматичне виконання тесту;
- опція [3] – «Setup Alert»: дозволяє налаштувати параметри для сповіщень.

Користувачу пропонується ввести номер опції, щоб виконати відповідну дію. Цей тип інтерфейсу є типовим для консольних програм, де взаємодія з користувачем відбувається через текстові команди.

При виконанні команди «Run Test», програма ініціює процес тестування, після чого відобразить графік у вікні GnuPlot, а також результати у консолі, що можна побачити на рисунку 4.2.

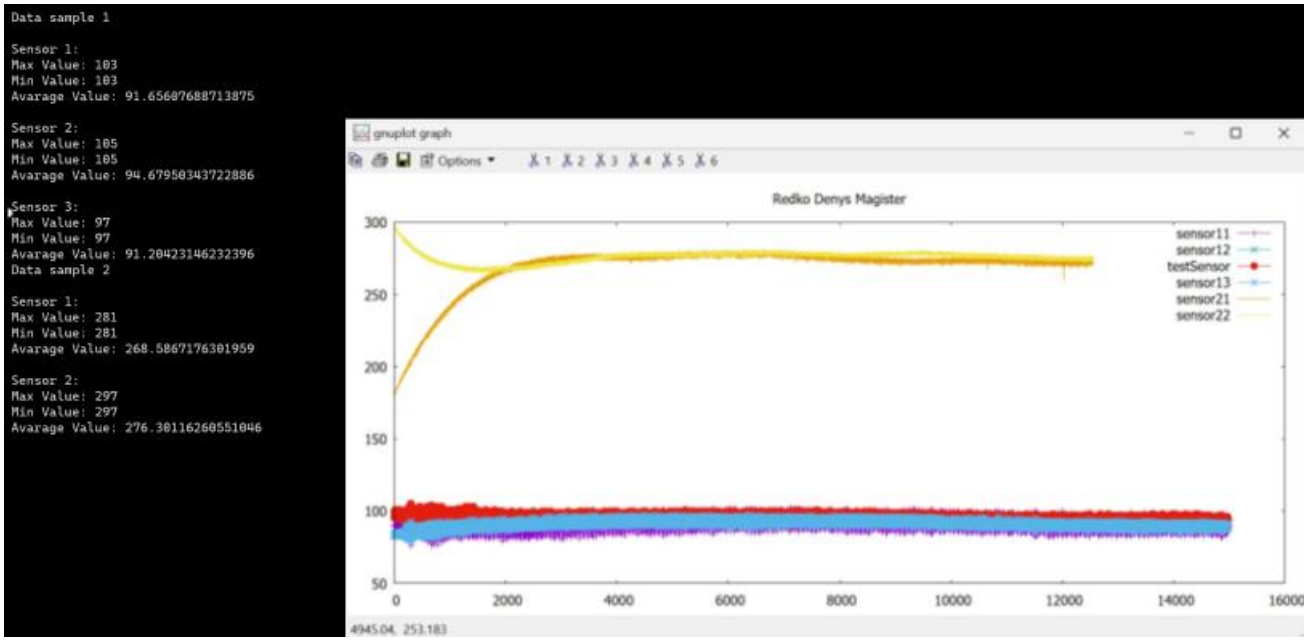


Рисунок 4.2 – Вивід результату тесту

При виконанні команди «Show Last Results», програма виводить список тестів, які були виконані. Результати цих тестів можна переглянути написавши у консоль порядковий номер тесту, що модна побачити на рисунку 4.3. Вивід самих результатів ідентичний до рисунку 4.2.

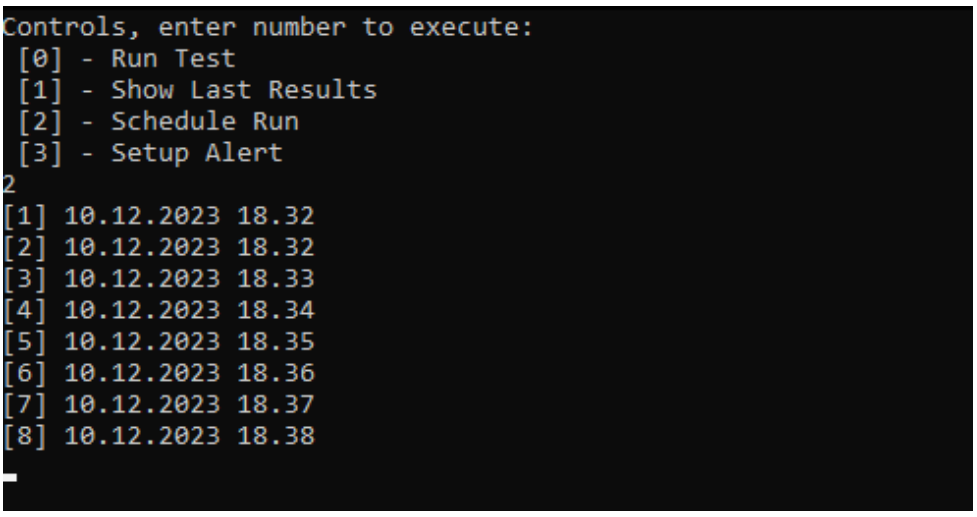


Рисунок 4.3 – Вивід списку тестів

4.2 Огляд можливостей системи для автоматизованого тестування

Також програма наділена можливостями для автоматизації процесу аналізу, до них входят пунки «Schedule Run» та «Setup Alert».

При виконанні команди «Schedule Run», програма пропонує ввести часовий проміжок у хвилинах, після вводу значення, через кожний зазначений проміжок часу, тест буде проходити автоматично. Якщо значення 0, то функція вимкнена. Приклад інтерфейсу можна поглянути на рисунку 4.4.

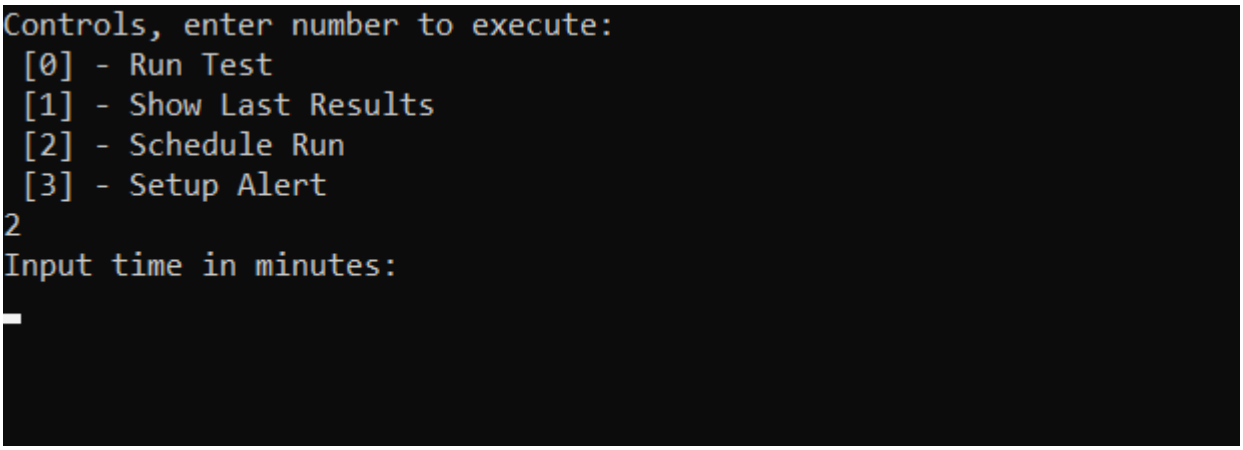


Рисунок 4.4 – Функція автоматичного виконання тесту

Коли користувач обирає команду «Setup Alert», програма надає можливість встановити два критичні параметри: максимальне та мінімальне значення. Ці значення використовуються для моніторингу результатів аналізу даних. Після введення цих порогових значень, програма автоматично генерує сповіщення, коли актуальні показники виходять за ці межі. Приклад інтерфейсу, можна побачити на рисунку 4.5.

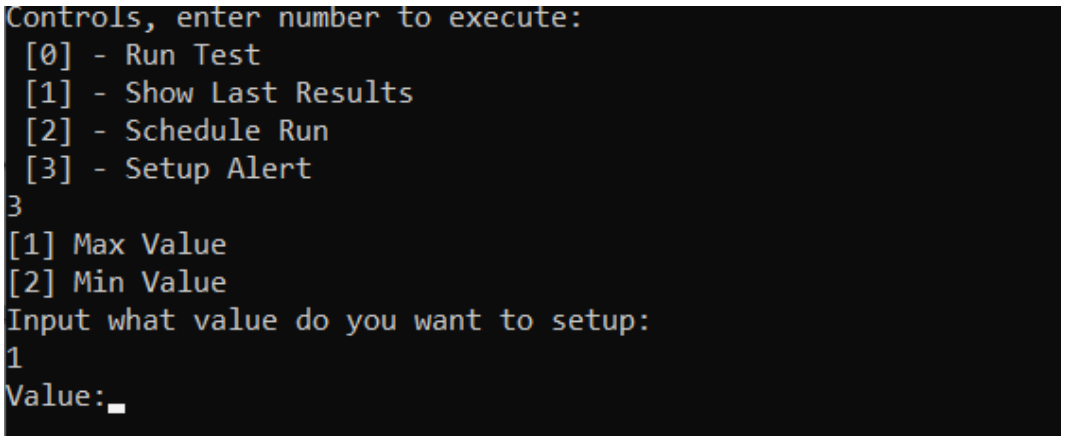


Рисунок 4.5 – Функція автоматичного виконання тесту

Таким чином була розглянута функціональність розробленої системи та надано детальну інструкцію користувачу.

Висновки до розділу 4

В даному розділі була розглянута та продемонстрована вся функціональність розробленої системи та надана детальна інструкція користувача. Система повністю вирішує поставлені задачі та може використовуватись в реальних бізнес задачах. Детальна інструкція користувача забезпечує легке та зрозуміле введення в експлуатацію системи, дозволяючи операторам швидко освоїти управління тестуванням, перегляд результатів, планування перевірок та налаштування сповіщень. Це не тільки сприяє збільшен ню продуктивності роботи операторів, а й підвищує загальну безпеку інфраструктурних об'єктів за рахунок своєчасного реагування на потенційні проблеми.

Таким чином, реалізована система є високоадаптивною та здатною задовольнити потреби сучасного моніторингу інфраструктури, гарантуючи точність, оперативність виявлення змін та простоту управління процесами моніторингу.

5 РОЗРОБКА СТАРТАПУ ПРОЄКТА

В даному розділі буде проведений маркетинговий аналіз стартап проєкту, розглянуті можливості ринкового впровадження системи, проблеми, які можуть виникнути при виході на «ринок».

5.1 Описання ідеї проєкту

Розглянемо зміст ідеї проєкту, що пропонується, можливі напрямки застосування та вигоди для користувачів в таблиці 5.1.

Таблиця 5.1 – Описання ідеї проєкту

Зміст ідеї	Напрямок застосування	Вигоди для напрямку
1	2	3
Розробка пристрою на базі Arduino для неруйнівного контролю металевих конструкцій.	Промисловість	Можливість швидкої оцінки стану великих промислових об'єктів, що забезпечує економію часу та підвищення безпеки на виробництві.
	Авіація	Забезпечення точного моніторингу стану літаків та їх компонентів для попередження аварійних ситуацій та підвищення безпеки польотів.

Продовження таблиці 5.1

1	2	3
	Суднобудування	Ефективний контроль цілісності суден, виявлення потенційних дефектів конструкції, зниження ризику аварій на морі.
	Залізнична галузь	Використання для контролю стану залізничних рейок та вагонів, зменшення вірогідності аварій через знос та пошкодження.
	Будівництво	Моніторинг стану будівельних конструкцій, виявлення дефектів на ранніх стадіях, що дозволяє уникнути серйозних пошкоджень і витрат на ремонт.
	Космічна галузь	Надійний контроль стану космічних

Продовження таблиці 5.1

1	2	3
	Нафтохімічна галузь	Ефективний моніторинг стану обладнання та трубопроводів, запобігання витоків та аварій.

Ця деталізована таблиця відображає різноманітність потенційних застосувань розробленого пристрою і вказує на специфічні вигоди для кожного з цих секторів.

Окрім того, для оцінки конкурентоспроможності проекту необхідно провести аналіз технічних та економічних переваг задуму. Це включає порівняння створеної системи з продуктами конкурентів, виявлення її міцних та слабких моментів, а також розробку стратегії для зменшення впливу недоліків та максимального використання переваг. Детальне порівняння представлене у таблиці 5.2.

Таблиця 5.2 – Визначення сильних, слабких та нейтральних характеристик

Характеристика	Розроблена система	Конкуренти	Результат
1	2	3	4
Гнучкість системи	Використання Arduino дозволяє легко модифікувати компоненти	Більшість конкурентних системвикористовують закриті платформи, що обмежує гнучкість.	Сильна

Продовження таблиці 5.2

1	2	3	4
Вартість реалізації	Завдяки використанню доступної Arduino-платформи та стандартних компонентів вартість реалізації низька.	Конкурентні системи часто використовують дорожчі компоненти та складніші технології, що підвищує вартість.	Сильна
Точність моніторингу	Точність моніторингу слабша ніж у конкурентів за рахунок масштабу тестування і дешевих сенсорів.	Точність конкурентних систем також висока, але може варіюватися залежно від моделі та виробника.	Слабка
Складність використання	Необхідні технічні знання для ефективного використання системи на базі Arduino.	Конкурентні системи часто мають більш інтуїтивний інтерфейс та меншу технічну складність.	Слабка

Продовження таблиці 5.2

1	2	3	4
Обсяг моніторингу	Можливість огляду всієї конструкції, а не тільки окремих її частин.	Більшість конкурентних систем фокусуються на детальному скануванні окремих ділянок.	Сильна
Вдосконалення та модифікації	Легке оновлення та адаптація завдяки відкритій платформі та широкому співтовариству Arduino.	Оновлення та модифікації в конкурентних системах можуть бути обмеженими або потребувати спеціалізованих знань.	Сильна
Швидкість обробки даних	Висока швидкість обробки даних завдяки ефективній архітектурі та використанню системи реального часу.	Швидкість обробки даних у конкурентів може бути різною, в залежності від специфікацій та архітектури системи.	Нейтральна

Ця таблиця демонструє, чи має розроблена система переваги перед конкурентами за основними характеристиками таких систем. В стовпці «Результат» вказано, що якщо там «Сильна», то у системи є перевага у відповідній

характеристиці порівняно з конкурентами. «Слабка" означає відсутність переваги, а «Нейтральна» свідчить про те, що системи майже ідентичні за цією характеристикою. Згідно з таблицею, система має як сильні, так і слабкі сторони, проте сильних аспектів більше, тому варто розглянути можливість реалізації стартап-проекту.

5.2 Технологічний аудит ідеї проєкту

Необхідно оцінити, чи є доступні необхідні технології для запуску проекту, та виробити узагальнення стосовно перспектив започаткування стартапу, враховуючи використовувані технології. Відповідні дані та аналіз знаходяться у таблиці 5.3.

Таблиця 5.3 – Технологічна здійсненність стартап проєкту

Складова проєкту	Технології реалізації	Наявність технологій	Доступність технологій
1	2	3	4
Моніторинг стану металевих конструкцій	П'єзоелектричні сенсорина для Arduino	Наявні	Широко доступні, недорогі (Arduino Uno –близько $25)
Апаратні контролери	Arduino, Raspberry Pi	Наявні	Доступні, Arduino, Raspberry Pi (35-50$)

Продовження таблиці 5.3

1	2	3	4
Обробка та аналіз даних	C#, .NET Framework, алгоритми машинного навчання	Наявні	C# – безкоштовний, деякі фреймворки можуть бути платними
Візуалізація даних	GnuPlot, графічні інтерфейси користувача	Наявні	GnuPlot – безкоштовний, інші інтерфейси – різна вартість
Зберігання даних	СУБД (напр., PostgreSQL, MongoDB)	Наявні	Безкоштовні або умовно безкоштовні версії доступні
Генератори вібрацій	Електромеханічні генератори	Наявні	Різноманітні моделі доступні, вартість залежить від специфікацій
Сервіси для сповіщення	SNS (Amazon Simple Notification Service)	Наявні	Доступний, вартість залежить від використання

З аналізу таблиці видно, що всі необхідні технології для запуску стартапу доступні. Більшість з них можна використовувати безкоштовно, хоча деякі з них є платними. Це дозволяє зробити висновок про те, що реалізація стартап проєкту є цілком можливою.

5.3 Аналіз ринкових можливостей запуску стартап проєкту

Перед запуском стартапу важливо провести аналіз ринкових можливостей, які можуть сприяти успіху проекту, та потенційних ризиків, що можуть завадити його реалізації. Важливо також врахувати загальний стан ринку, потреби потенційних клієнтів, а також діяльність і пропозиції конкурентів. Детальніший аналіз ринку та його характеристики будуть викладені у таблиці 5.4.

Таблиця 5.4 – Характеристика ринку дефектометрів

Кількість конкурентів	Понад 100 компаній
Загальний обсяг продажів	Вимірюється мільйонами доларів
Динаміка ринку	Позитивна зі зростанням CAGR на 5.6% з 2023 по 2030
Наявність обмежень для входу	Високі, через концентрацію ринку в руках великих гравців
Вимоги до специфікації	Рекомендовано використання стандартів якісної сертифікації
Середня норма рентабельності	Орієнтовно 35% за ринковими оцінками

Ця таблиця, що відображає стан ринку ультразвукових дефектоскопів, надає ключову інформацію про конкурентне середовище та можливість входу на

ринок для нових гравців. Висока кількість існуючих конкурентів та значний обсяг продажів свідчать про зрілість та насиченість ринку, який оцінюється мільйонами доларів. Динаміка ринку позитивна зі стабільним зростанням, що може бути заманливим для нових компаній, але наявність високих бар'єрів для входу, які створюються домінуючими гравцями ринку, може стати перешкодою для новачків.

Отже, для нових гравців, які бажають увійти на ринок ультразвукових дефектоскопів, буде важливим зрозуміти та адаптуватися до існуючих ринкових умов, знайти унікальні конкурентні переваги та розробити стратегії для преодолення бар'єрів. Визначимо потенційних клієнтів, їхні характеристики та їхні вимоги на таблиці 5.5.

Таблиця 5.5 – Характеристика потенційних клієнтів

Потенційний клієнт	Потреби	Очікування
1	2	3
Промислові підприємства	Висока точність виявлення дефектів для запобігання зупинок виробництва та аварій.	Надійне обладнання, яке може працювати безперебійно в продуктивному промисловому середовищі.
Енергетичні компанії	Портативність для легкості перенесення між об'єктами та легкість використання в полевих умовах.	Технічна підтримка та обслуговування, щоб забезпечити неперервність енергопостачання.

Продовження таблиці 5.5

1	2	3
Будівельні організації	Здатність виявляти дефекти в товстих бетонних конструкціях і залізобетонних елементах.	Здатність до швидкої калібрування та налаштування під різні будівельні матеріали.
Авіаційна галузь	Високі стандарти точності та чутливості для забезпечення безпеки польотів.	Сертифікація обладнання згідно з авіаційними регуляторними вимогами.
Нафтогазовий сектор	Опірність до екстремальних умов (температура, тиск) для використання на відкритих морських платформах.	Водо- та ударостійкість для використання в суворих умовах експлуатації.
Суднобудування	Інтеграція з існуючими системами діагностики та технічного обслуговування кораблів.	Міцність т у морських умовах, легкість у обслуговуванні та ремонті.
Залізничний транспорт	Довговічність датчиків для виявлення дефектів на рейках, що забезпечує безпеку руху поїздів.	Економічна ефективність для забезпечення експлуатації залізниць.

Були розглянуті потенційні клієнти, їхні потреби та очікування, щоб розуміти наскільки реалізована система їх задовольняє, на на чому потрібно сконцетруватись та покращити. Цей аналіз допомагає визначити ключові напрямки для розробки та маркетингу ультразвукових дефектоскопів, орієнтуючись на конкретні потреби кожного сегменту ринку, основне завдання полягає в тому, щоб забезпечити, що продукт відповідає цим вимогам, що, у свою чергу, сприятиме успішному входу на ринок і залученню нових клієнтів. Також необхідно розглянути фактори загроз та можливостей, що наведено в таблицях 5.6 та 5.7.

Таблиця 5.6 – Фактори загроз стартап проєкту

Фактор	Зміст загрози	Реакції клієнта
1	2	3
Висока конкуренція	Стартап може зіткнутися з великою кількістю вже існуючих, сильних гравців на ринку.	Клієнти можуть віддавати перевагу продуктам відомих брендів.
Обмежений доступ до фінансування	Брак фінансових ресурсів для розвитку та масштабування проекту.	Потенційний недолік інвестицій може зменшити довіру до стартапу.
Технологічні зміни	Швидкий розвиток технологій може зробити продукт стартапу застарілим.	Клієнти можуть шукати більш інноваційні або оновлені рішення.

Продовження таблиці 5.6

1	2	3
Вимоги законодавства	Строгі регуляторні вимоги та зміни у законодавстві.	Юридичні перешкоди можуть викликати занепокоєння або недовіру клієнтів.
Проблеми з якістю	Можливість випуску продукту з низькою якістю або дефектами.	Невдоволення клієнтів та втрата репутації на ринку.
Зміни у споживацьких уподобаннях	Нездатність адаптуватися до змін у вимогах та бажаннях клієнтів.	Клієнти можуть перейти до конкурентів, які краще задовольняють їх потреби.

Таблиця 5.7 – Фактори можливостей стартап проєкту

Фактор	Можливості	Вплив на ринок
1	2	3
Інноваційність продукту	Унікальна розробка, якої немає у конкурентів.	Висока цінність пропозиції на ринку, можливість встановлення трендів.
Вдосконалення технологій	Постійне оновлення та вдосконалення продукту.	Збільшення довіри клієнтів, репутація інноватора.

Продовження таблиці 5.7

1	2	3
Нішевий ринок	Фокусування на специфічних потребах нішевих сегментів.	Можливість стати лідером у ніші.
Адаптивність до ринку	Здатність швидко адаптуватися до змін ринкових умов.	Збереження конкурентоспроможності в довгостроковій перспективі.

Фінальним кроком у процесі аналізу ринкових можливостей є створення SWOT-аналізу. Це означає формування матриці, яка включає оцінку сильних та слабких сторін, а також виявлення можливостей і потенційних загроз для проекту. Дану матрицю можна знайти у таблиці 5.8, де вона надає цілісний огляд ключових аспектів, що впливають на успіх чи провал стартапу.

Таблиця 5.8 – Матриця SWOT аналізу

Категорія	Опис
Сильні стороні	Унікальний продукт без прямих конкурентів на ринку; висока точність та надійність; інноваційність технології.
Слабкі стороні	Великі бар'єри для входу на ринок; обмежений доступ до фінансування; потреба у висококваліфікованих фахівцях.
Можливості	Зростаючий попит на високоточне обладнання; розширення на нові ринки; потенціал для співпраці з великими брендами.
Загрози	Швидкий розвиток технологій може зробити продукт застарілим; висока конкуренція у секторі; зміни в законодавстві.

Висновки до розділу 5

На основі проведеного маркетингового аналізу для стартап проекту ультразвукових дефектоскопів можна зробити такі висновки:

- ринкові можливості: проект має унікальну пропозицію на ринку завдяки високій точності та інноваційності своєї розробки. Відсутність прямих конкурентів є значною перевагою, що дозволяє зайняти нішу на ринку. Зростаючий попит на високоточне обладнання в різних галузях, таких як авіація, енергетика, нафтогазовий сектор, створює додаткові можливості для розширення ринку;
- проблеми при виході на ринок: незважаючи на унікальність продукту, стартап може зіткнутися з викликами, пов'язаними з перешкодами при вході на ринок, обмеженим доступом до фінансування та необхідністю відповідності продукту строгим законодавчим стандартам. Швидкий розвиток технологій також може стати загрозою, оскільки потребує постійної адаптації та оновлення продукту, щоб залишатися конкурентоспроможним;
- маркетингова стратегія: важливим фактором успіху буде розробка ефективної маркетингової стратегії, що зосереджується на унікальних перевагах продукту та вибудовування міцних партнерських відносин з ключовими гравцями на ринку. Також необхідно звернути увагу на налагодження зв'язків з потенційними клієнтами та розробку адаптивних рішень, які задовольняють їх конкретні потреби.

У підсумку, проект має значний потенціал для успішного входу на ринок, але потребує ретельного планування та адаптації до його динаміки та викликів.

ВИСНОВКИ

Була виконана магістерська дисертація, в рамках якої був реалізований пристрій і програмне забезпечення для моніторингу стану металевих конструкцій. Основна мета цього пристрою полягає в оперативному контролі структур, своєчасному виявленні потенційних проблем і мінімізації ризиків пошкодження або руйнації металевих конструкціях. Система аналізує дані вібраційних сенсорів, що дозволяє виявляти зміни в структурі металу.

В роботі було використано апаратне та програмне забезпечення на базі Arduino. Використання Arduino забезпечило гнучкість та доступність у розробці пристрою. П'єзоелектричні сенсори та генератори вібрацій використовувалися для точного вимірювання вібрацій в металевих конструкціях. П'єзоелектричні сенсори ефективні у перетворенні вібрацій на електричний сигнал, що дозволяє детально аналізувати стан конструкцій. Використання цих сенсорів разом з генераторами вібрацій дало змогу точно визначати потенційні проблеми та зміни в структурі металу, забезпечуючи високий рівень точності та надійності моніторингу.

Для обробки даних та візуалізації результатів моніторингу вібрацій, використовувалися .NET і C# у поєднанні з GnuPlot. .NET та C# забезпечували надійну платформу для розробки програмного забезпечення, що дозволяло ефективно обробляти великі обсяги даних. GnuPlot використовувався для створення графічних зображень вібраційних показників, що сприяло кращому розумінню та аналізу стану металевих конструкцій.

У моїй магістерській роботі розроблена система успішно вирішує поставлене завдання та відповідає всім необхідним вимогам, що робить її придатною для використання. Крім того, було проведено маркетинговий та ринковий аналіз з метою оцінки перспектив впровадження розробленої системи як стартап-проекту. Аналіз показав, що реалізація такого проекту є можливою, хоча і супроводжується певними складнощами та викликами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Global Ultrasonic Flaw Detector Market Report 2024 Market Size Split by Type. URL: https://www.cognitivemarketresearch.com/ultrasonic-flaw-detector-market-report (дата звернення: 24.12.2023).
2. Global Advanced Ultrasonic Flaw Detector Market Report 2024 Market Size. URL: https://www.cognitivemarketresearch.com/advanced-ultrasonic-flaw-detector-market-report (дата звернення: 24.12.2023).
3. Global Ultrasonic Flaw Detectors for Aerospace Market Report 2024 Market Size Split by Type. URL: https://www.cognitivemarketresearch.com/ultrasonic-flaw-detectors-for-aerospace-market-report (дата звернення: 24.12.2023).
4. Офіційна сторінка GE Measurement. URL: https://gms-instruments.com/brand/ge-measurement-control/ (дата звернення: 24.12.2023).
5. Офіційна сторінка Novotest. URL: https://www.novotest.ua/ua/ (дата звернення: 24.12.2023).
6. Офіційна сторінка Olympus. URL: https://www.olympus-ims.com/en/ut-flaw/ (дата звернення: 24.12.2023).
7. Офіційна сторінка Sonatest. URL: https://sonatest.com/ (дата звернення: 24.12.2023).
8. Офіційна сторінка Arduino. URL: https://www.arduino.cc/ (дата звернення: 24.12.2023).
9. Marlina Susanti, Meidi Arisalwadi, Menasita Mayantasari Prototype Design Of Geophone Using Piezoelectric Sensor Based On Arduino URL: https://jurnal.institutsunandoe.ac.id/index.php/prevenire/article/view/28 (дата звернення: 24.12.2023).
10. Active and Passive Vibration Control of Structuresю URL: https://www.researchgate.net/publication/321612454_Active_and_Passive_Vibration_Control_of_Structures (дата звернення: 24.12.2023).

11. Sajad Hadidi, Alireza Hassanzadeh A Novel Self-Powered, High-Sensitivity Piezoelectric Vibration Sensor Based on Piezoelectric Combo Effect IEEE Sensors Journal, 2023. p 99.
12. Changhwan Kim, Gyeonghwan Yun Analysis of Vibration and Noise in a Permanent Magnet Synchronous Motor Based on Temperature-Dependent Characteristics of Permanent Magnet Energies MDPI 16, 2023. p 16.
13. Masse Mark REST API Design Rulebook Publisher by O'Reilly Media 1st edition, 2011. p 112.
14. Freeman A. Pro ASP.NET Core 3. Apress, 2020. p 1400.
15. Introduction to ASP.NET Core. URL: https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-3.0. (дата звернення: 24.12.2023).
16. Документація C# . URL: https://learn.microsoft.com/uk-ua/dotnet/csharp/
17. C# In Depth 4 th edition, 2008. p 230.
18. Документація GnuPlot . URL: http://www.gnuplot.info/documentation.html
19. Документація Gnuplot CSharp. URL: https://github.com/AwokeKnowing/GnuplotCSharp (дата звернення: 24.12.2023).
20. GnuPlot In Action Book, 2016. p 315.
21. Minghao Chen, Qibo Mao Active Vibration Control of a Plate Using Piezoelectric Sensors and Loudspeakers / Proceedings of the 6th China Aeronautical Science and Technology Conference 2023, p 596.
22. Detect a Knock Tutorial. URL: https://docs.arduino.cc/built-in-examples/sensors/Knock (дата звернення: 24.12.2023).
23. Controlling a DC Motor Tutorial. URL: https://docs.arduino.cc/tutorials/motor-shield-rev3/msr3-controlling-dc-motor (дата звернення: 24.12.2023).
24. SNS documentation. URL: https://docs.aws.amazon.com/sns/latest/dg/welcome.html (дата звернення: 24.12.2023).

ДОДАТКИ

ДОДАТОК А Лістинг програми

Program.ino

```
// Визначення пінів для датчиків та моторчиків
const int sensorPin1 = A0;
const int sensorPin2 = A1;
const int sensorPin3 = A2;
const int motorPin1 = 3;
const int motorPin2 = 4;
const int motorPin3 = 5;

String inputString = "";         // Рядок для зберігання вхідних
даних
boolean stringComplete = false;  // Чи завершено ввод рядка

void setup() {
  // Ініціалізація серійного зв'язку
  Serial.begin(9600);

  // Налаштування пінів моторчиків як виходів
  pinMode(motorPin1, OUTPUT);
  pinMode(motorPin2, OUTPUT);
  pinMode(motorPin3, OUTPUT);

  // Очистка вхідного рядка
  inputString.reserve(200);
}

void loop() {
  // Чекаємо на команду "StartTest"
  if (stringComplete) {
    if (inputString == "StartTest") {
      // Виконання тесту
      performTest();
    }

    // Очищення вхідного рядка для наступного вводу
    inputString = "";
    stringComplete = false;
  }
}

void serialEvent() {
  while (Serial.available()) {
    // Отримання одного символа
    char inChar = (char)Serial.read();
```

```
    // Додавання його до вхідного рядка
    inputString += inChar;
    // Якщо отримано символ завершення рядка, встановлюємо флаг
завершення
    if (inChar == '\n') {
      stringComplete = true;
    }
  }
}

void performTest() {
  // Вмикання моторчиків
  digitalWrite(motorPin1, HIGH);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin3, HIGH);

  // Запуск таймера
  unsigned long startTime = millis();
  while (millis() - startTime < 10000) { // Програма працює 10
секунд
    // Зчитування значень з датчиків
    int sensorValue1 = analogRead(sensorPin1);
    int sensorValue2 = analogRead(sensorPin2);
    int sensorValue3 = analogRead(sensorPin3);

    // Виведення результатів у серійний монітор
    Serial.print("Sensor 1: ");
    Serial.print(sensorValue1);
    Serial.print("\tSensor 2: ");
    Serial.print(sensorValue2);
    Serial.print("\tSensor 3: ");
    Serial.println(sensorValue3);

    delay(100); // Затримка для уникнення засмічення серійного
монітора
  }

  // Вимикання моторчиків
  digitalWrite(motorPin1, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin3, LOW);
}
```

Program.cs

```
class Program
{
    static void Main(string[] args)
    {
        bool exitProgram = false;
        while (!exitProgram)
```

```
        {
            Console.WriteLine("[0] - Run Test");
            Console.WriteLine("[1] - Show Last Results");
            Console.WriteLine("[2] - Schedule Run");
            Console.WriteLine("[3] - Setup Alert");
            Console.WriteLine("[4] - Exit");

            Console.Write("Виберіть опцію: ");
            string option = Console.ReadLine();

            switch (option)
            {
                case "0":
                    RunTest();
                    break;
                case "1":
                    ShowLastResults();
                    break;
                case "2":
                    ScheduleRun();
                    break;
                case "3":
                    SetupAlert();
                    break;
                case "4":
                    exitProgram = true;
                    break;
                default:
                    Console.WriteLine("Неправильний вибір.
Спробуйте ще раз.");
                    break;
            }
        }
    }

    static void RunTest()
    {
        string portName = "COM3"; // Замініть на відповідний порт
        int baudRate = 9600; // Переконайтеся, що цей параметр
відповідає налаштуванням у вашому Arduino коді

        using (SerialPort serialPort = new SerialPort(portName,
baudRate))
        {
            try
            {
                // Відкриття порту
                serialPort.Open();

                // Надсилання команди "StartTest" на Arduino
                serialPort.WriteLine("StartTest");
```

```
                Console.WriteLine("Команда 'StartTest' надіслана до
Arduino.");

                // Чекаємо деякий час для отримання відповіді
                Thread.Sleep(2000); // Чекаємо 2 секунди

                // Читання даних з серійного порту
                string responseData = ReadSerialData(serialPort);
                Console.WriteLine("Отримані дані: " +
responseData);

                // Збереження даних у файл
                SaveDataToFile(responseData);
                var analyzeData = AnalyzeSensorData(responseData);
                SaveAnalyzeDataToFile(analyzeData);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Помилка: " + ex.Message);
            }
        }
    }


    static IEnumerable<SensorData> AnalyzeSensorData(string input)
    {
        // Розділення вхідного тексту на рядки
        string[] lines = input.Split(new[] { '\n' },
StringSplitOptions.RemoveEmptyEntries);

        // Обчислення кількості сенсорів
        int numberOfSensors = lines[0].Split(',').Length;
        double[] sum = new double[numberOfSensors];
        double[] max = Enumerable.Repeat(double.MinValue,
numberOfSensors).ToArray();
        double[] min = Enumerable.Repeat(double.MaxValue,
numberOfSensors).ToArray();

        int count = 0;
        foreach (var line in lines)
        {
            // Розділення кожного рядка на значення
            var values =
line.Split(',').Select(double.Parse).ToArray();

            for (int i = 0; i < numberOfSensors; i++)
            {
                sum[i] += values[i];
                max[i] = Math.Max(max[i], values[i]);
                min[i] = Math.Min(min[i], values[i]);
            }
            count++;
```

```
        }

        // Обчислення середніх значень
        double[] average = sum.Select(s => s / count).ToArray();
        var dataToReturn = new IEnumerable<SensorData>();
        // Виведення результатів
        for (int i = 0; i < numberOfSensors; i++)
        {
             dataToReturn.Add(new SensorData(max[i], min[i],
average[i]))
        }
    }

    static string ReadSerialData(SerialPort serialPort)
    {
        string data = "";
        while (serialPort.BytesToRead > 0)
        {
            data += serialPort.ReadLine() + "\n";
        }
        return data;
    }

    static void SaveDataToFile(string data)
    {
        string fileName = "TestData_" +
DateTime.Now.ToString("yyyyMMdd_HHmmss") + ".txt";
        File.WriteAllText(fileName, data);
        Console.WriteLine("Дані збережені у файлі: " + fileName);
    }

    static void ShowLastResults()
    {
        Console.WriteLine("Відображення останніх результатів.");
         // Перевірка, чи існує директорія
        if (!Directory.Exists(RESULT_FOLDER))
        {
            Console.WriteLine("Директорія не знайдена: " +
RESULT_FOLDER);
            return;
        }

        // Отримання всіх файлів у директорії
        string[] files = Directory.GetFiles(RESULT_FOLDER);

        foreach (string file in files)
        {
            // Виведення імені файлу
            Console.WriteLine("Файл: " + file);

            // Читання та виведення вмісту файлу
            try
```

```
            {
                string content = File.ReadAllText(file);
                Console.WriteLine("Вміст файлу:\n" + content);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Не вдалося прочитати файл: " +
ex.Message);
            }

            Console.WriteLine(); // Додавання пустої лінії для
кращої читабельності
        }
    }
}
```

SensorData.cs

```
public class SensorData
{
    public double MaxValue { get; set; }
    public double MinValue { get; set; }
    public double AverageValue { get; set; }

    public SensorData(double maxValue, double minValue, double
AverageValue)
{
     MaxValue = maxValue;
MinValue = minValue;
AverageValue = averageValue;
}
}
```

SensorDataProcessor.cs

```
public class SensorDataProcessor

{

    private double MaxLimit; // Приклад обмеження

    private double MinLimit; // Приклад обмеження

    private AmazonSimpleNotificationServiceClient snsClient;


    public SensorDataProcessor(ILimitsProvider _limitProvider)

    {

        snsClient = new AmazonSimpleNotificationServiceClient();
```

```
     MaxLimit = _limitProvider.Max;
     MinLimit = _limitProvider.Min;
 }

    public async Task CheckAndNotifyAsync(IEnumerable<SensorData>
sensorDataList)
    {
        foreach (var data in sensorDataList)
        {
            if (data.MaxValue > MaxLimit || data.MinValue <
MinLimit)
            {
                await NotifyAsync($"Виявлено відхилення: Max:
{data.MaxValue}, Min: {data.MinValue}, Avg: {data.AverageValue}");
            }
        }
    }

    private async Task NotifyAsync(string message)
    {
        var request = new PublishRequest
        {
            TopicArn = Topic ARN,
            Message = message
        };

        await snsClient.PublishAsync(request);
    }
}
```

TestRunner.cs

```
public class TestRunner
{
    private Timer _timer;
```

```
    public void ScheduleRun(int interval)
    {
        // Встановлення таймера
        _timer = new Timer(RunTest, null, 0, interval);
    }
}
```

ДОДАТОК Б Презентація

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Кафедра Інженерії програмного забезпечення в енергетиці.

Реєстратор вібрації поверхні на базі апаратно-програмних засобів Arduino

Виконав: студент магістерського рівня 2-го року навчання,
групи ТВ-22 МП
Редько Денис Вадимович
Керівник: д.т.н. Федорова Н. В.

АКТУАЛЬНІСТЬ ТЕМИ ДОСЛІДЖЕННЯ

Ріст нашого сучасного суспільства неможливий без ефективної інфраструктури, яка включає в себе мости, будівлі, інженерні споруди та інші металеві конструкції. Вони створюють основу для безпеки та комфорту мільйонів людей щодня. Проте ці конструкції піддаються впливу різних факторів, таких як вібрації від транспорту та природних явищ, навантаження і знос від корозії. Існує постійне завдання контролювати стан конструкцій із металу, аби попереджати можливі пошкодження та деформації.

Моніторинг стану інфраструктури за допомогою сенсорів коливань на основі системи Arduino

ЗАДАЧА МОНІТОРИНГУ СТАНУ ІНФРАСТРУКТУРИ

Мета: розробка програмного і апаратного продукту для моніторингу за металевими конструкціями

Об'єкт дослідження: алгоритм порівняння даних з сенсорів коливань

Предмет дослідження: програмний і апаратний продукт для моніторингу за структурними змінами у металевих конструкціях

Задачі, які потрібно розв'язати для досягнення мети:

- проаналізувати методи моніторингу за структурними змінами у металевих конструкціях;
- розробити алгоритм порівняння аналогових даних з сенсорів;
- розробити програмне рішення для збору інформації про стан об'єкту;
- спроектувати архітектуру програмних компонентів;
- розробити програмне забезпечення для візуалізації та контролю за збором даних.

ІСНУЮЧІ АНАЛОГИ

Eddyfi Technologies

NOVOTEST

OKOndt GROUP

ЗАДАЧІ ЗБОРІ ДАНИХ ТА ЇХ ПОРІВНЯННЯ ДЛЯ ВИЯВЛЕННЯ ВІДХИЛЕНЬ

Збір даних

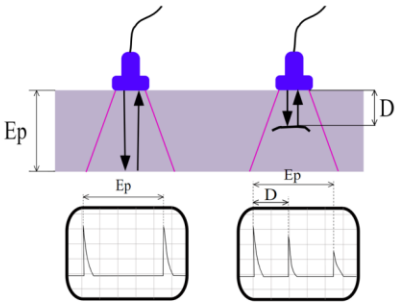
Метою є збір та збереження даних про поточний стан конструкції методом ультразвукової дефектоскопії.

Порівняння даних.

Метою є порівняння поточного стану з минулими даними для виявлення змін у показниках

УЛЬТРОЗВУКОВА ДИФЕКТОСКОПІЯ

Ультразвукова дефектоскопія - методів неруйнівного контролю що дозволяє здійснювати пошук дефектів в матеріалі виробу шляхом випромінювання та прийняття ультразвукових коливань, відбитих від внутрішніх несплошностей (дефектів)



АРХІТЕКТУРА ПРОГРАМНИХ ТА АПАРАТНИХ КОМПОНЕНТІВ

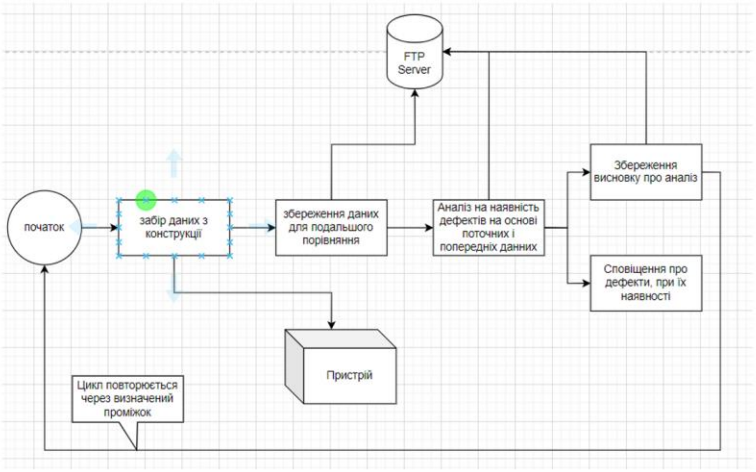
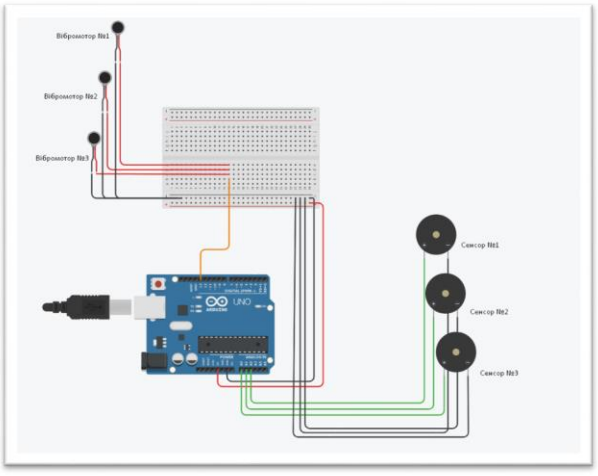


СХЕМА АПАРАТНОЇ ЧАСТИНИ НА ОСНОВІ ARDUINO



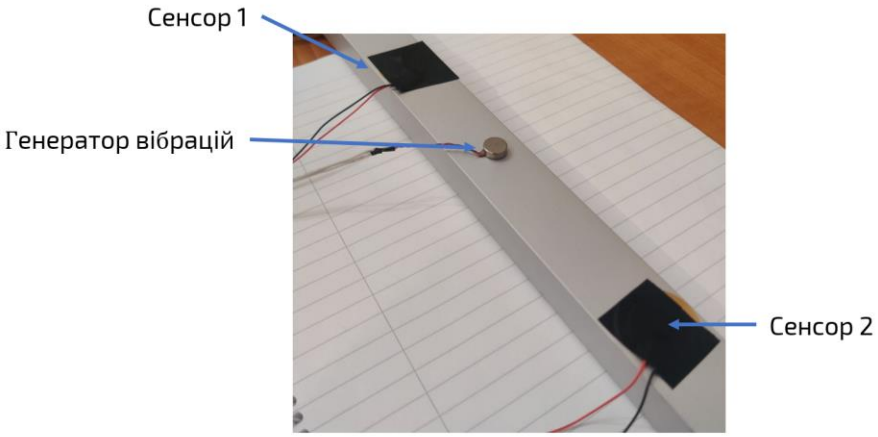
ЗАСОБИ РОЗРОБКИ

ARDUINO

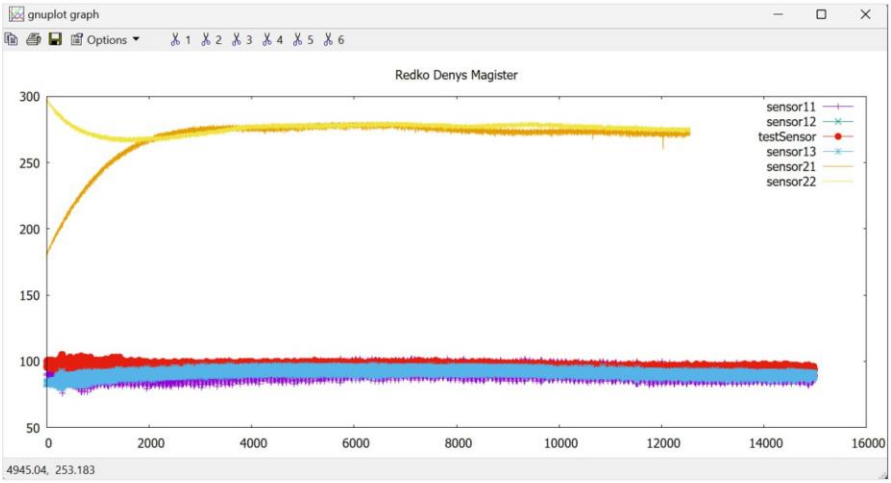
.NET

Gnuplot Scripts

ПРИКЛАД РОЗМІЩЕННЯ СЕНСОРІВ НА МЕТАЛЕВІЙ КОНСТРУКЦІЇ



ІНТЕРФЕЙС ДЛЯ ПОРІВНЯННЯ ГРАФІКІВ



ІНТЕРФЕЙС ДЛЯ ПОРІВНЯННЯ ЗНАЧЕНЬ

Senor 1	Senor 1
Mix value: 70	Mix value: 150
Max Value: 103	Max Value: 273
Average Value: 91	Average Value: 190
Senor 2	Senor 2
Mix value: 75	Mix value: 160
Max Value: 111	Max Value: 270
Average Value: 97	Average Value: 181
Senor 3	Senor 3
Mix value: 73	Mix value: 168
Max Value: 112	Max Value: 274
Average Value: 96	Average Value: 190

ВИСНОВКИ

Задачі, які було розв’язано для досягнення мети:

- проаналізовано методи моніторингу за структурними змінами у металевих конструкціях;
- розроблено алгоритм порівняння аналогових даних з сенсорів;
- розроблено програмне рішення для збору інформації про стан об’єкту;
- спроектовано архітектуру програмних компонентів;
- розроблено програмне забезпечення для візуалізації та контролю за збором даних.

ДОДАТОК В Публікація

http://www.moderntechno.de/index.php/meit/article/view/meit21-01-010
DOI: 10.30890/2567-5273.2022-21-01-010

UDC 004.2

SURFACE VIBRATION RECORDER BASED ON ARDUINO HARDWARE AND SOFTWARE

ПОВЕРХНЕВИЙ ВІБРАЦІЙНИЙ РЕЄСТРАТОР НА ОСНОВІ АПАРАТНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ARDUINO

Fedorova N.V. / Федорова Н.В.
d.t.s. / д.т.н. доцент, профессор кафедры автоматизации проектирования энергетических процессов и систем
ORCID: 0000-0002-4548-4198
Redko D.V. / Редько Д.В.
undergraduate/ магістрант
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Prosp. Peremohy 37, 03056
Національний технічний університет України "Київський політехнічний інститут им. Ігоря Сікорського", Київ, пр-т Перемоги, 37

Аннотация. *Наукове дослідження на тему реєстрація вібрацій на поверхні за допомогою апаратно обчислювальної платформа для аматорського конструювання Ардуіно розкриває сучасні способи створення фізичного продукту для роботи з різними видами коливань на поверхні. Система налічує багато сенсорів сумісних з платформою, та компонентів які можна легко адаптувати для роботи, завдяки платі мікроконтролера з елементами вводу/виводу та середовищу розробки Processing/Wiring для Ардуіно. В роботі було поставлено таку задачу, як проведення трьох дослідів з трьома різними сенсорами у різних умовах, та дослідити здатність Ардуіно до адаптації під різні задачі та оточення, а також визначити можливості кожного з сенсорів. При проведенні дослідження було виконано три тести для збору даних про роботу сенсорів, які знаходилися у однакових умовах. Метою проведеної роботи є виявлення ідеальних умов та визначення технічних здібностей кожного з сенсорів, для цього умови у кожному тесті були різні, також доведення доцільності використання системи Ардуіно для створення прототипів. Результатами проведеного дослідження встановлено, що універсального датчика для вимірювання коливань на поверхні не існує, а кожен з представлених сенсорів підійде для окремих амплітудних діапазонів та специфічних цілей. З проведеного дослідження випливає висновок, що при створенні продукту, який має здібність, або необхідність до виміру коливань на поверхні, необхідно розуміти з якими саме коливаннями буде працювати пристрій, і саме це грає ключову роль у виборі правильного сенсора для системи Ардуіно.*

Ключові слова*: Ардуіно, коливання, сенсор вібрацій.*

1. Вступ

Коливальні процеси характерні для великої кількості природних та штучних процесів, які оточують сучасну людину. Саме робота з коливаннями, їх відтворення та вимірювання, дозволило людям так пришвидшити прогрес. У цій статті було розглянуто способи та різноманітні датчики для вимірювання коливань, різної сили, на поверхні.

Вимірювання коливань на поверхні має багато сфер застосування, починаючи з передбачення природних катастроф, закінчуючи перетворенням будь-якої поверхні на контактний інтерфейс для користувача. Система Arduino, полегшує розробку прототипів та мінімального робочого продукту для представлення інвесторам. Створену на основі Arduino плату можливо за короткий термін перетворити на самостійний пристрій, тому що Arduino має

відкритий код і схему, що дозволяє прибрати усе зайве, залишивши необхідне для правильної роботи прототипу. Сучасні підходи до розробки інформаційних технологій вимагають використання концептуального моделювання та спеціальних інструментальних засобів на етапі будування прототипу, щоб максимально зменшити витрати та пришвидшити створення мінімально дійочого продукту, який зможе довести можливість створення та використання продукту. Апаратна обчислювальна платформа Arduino для аматорського конструювання, допомагає інженерам у створенні прототипів.

2. Опис сенсорів які беруть участь у дослідженні

2.1. SW-520D – Датчик вібрації і нахилу

Датчик Sw-520D являє собою компактний датчик для ідентифікації нахилу, також може застосовуватися для вимірювання коливань на поверхні до якої прилягає пристрій. Конструкція складається з двох кульок поміщених у циліндр, які контактують з двома контактами пристрою якщо датчик перевищує певний кут, і навпаки [1]. Тільки коли датчик нахилу знаходиться у вертикальному положенні, металеві кульки всередині датчика нахилу з'єднують два контакти, завершуючи ланцюг. Коли датчик нахиляється за межі діапазону чутливості, контакти віддаляються, і таким чином розмикається ланцюг., при зміні кута чи при ударі кульки змінюють положення у циліндрі, що спричинює розмикання ланцюга, на основі якого і робиться висновок про коливання. Через простоту конструкції датчика, точність його не дуже велика, цифровий комунікаційний вихід, не дає змогу отримувати данні про силу коливань а лише про їх наявність, що унеможливлює використання датчика для складних вимірювань, проте затримка у 2 мілісекунди дозволяє використовувати його у системах, де швидкість оклику критично важлива.

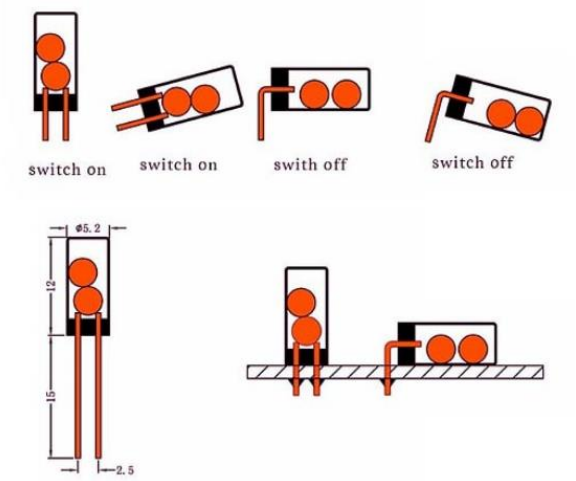


Рисунок 1 - Схема датчика SW-520D.

2.2. MPU-6050 – Акселерометр і гіроскоп

Датчик MPU-6050 (GY-521) модуль 3-х осьового акселерометра і 3-х осьового гіроскопа, а також температури. Гіроскоп вимірює швидкість

обертання або швидкість зміни кутового положення у часі по осях X, Y та Z [2]. Акселерометр надає можливість вимірювати гравітаційне прискорення по трьох осях, і використовуючи деяку тригонометрію, можна обчислити кут, під яким розташований датчик [3]. Отже, якщо поєднати дані акселерометра та гіроскопа, можливо отримати дуже точну інформацію про орієнтацію датчика в просторі. Датчик має можливість налаштування діапазону гіроскопу від ±250°/с до ±2000°/с та акселерометру від ±2g до ±16g, також передбачено багато інших налаштувань та вбудованих функцій, точність датчика гіроскопу здебільшого залежить від процесу початкового калібрування [2].

Завдяки цій компактній платі можна вимірювати коливання різної величини, від пересування пристрою до легких ударів по поверхні на якій знаходиться датчик. Аналоговий комунікаційний вихід дозволяє отримувати данні, про силу та напрямок коливань, у реальному часі, що вигідно відрізняє цей датчик від розглянутого у 2.1 датчика SW-520D. Для точного вимірювання куту нахилу використовують базову тригонометрію та дискретне інтегрування кутової швидкості, для отримання більш точного куту нахилу, оскільки Arduino відкрите середовище, більшість необхідних формул уже описане у бібліотеках створених користувачами. Точне вимірювання коливань у меншій мірі потребує даних з гіроскопа і у більшій з акселерометра [4], але обчислення куту нахилу корисне для визначення у якій плоскості відбуваються коливання.

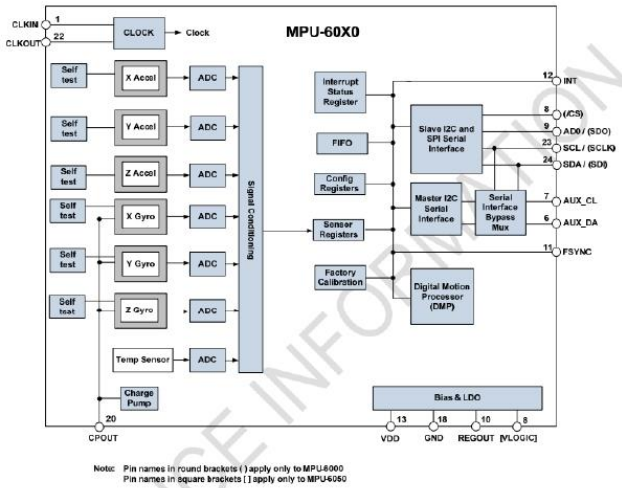


Рисунок 2 - Опис пінів для датчика MPU-60X0

2.3. Звукознімач п'єзоелектричний

П'єзоелектричні перетворювачі – пристрої, що використовують п'єзоелектричний ефект у кристалах, кераміці або плівках і перетворюють електричну енергію на механічну і навпаки.

Використовуючи кристалічну плівку, яку можна помістити на бажану поверхню, можна вимірювати коливання які впливають на п’єзоелемент. Такий датчик використовують у музичних інструментах, щоб перетворювати коливання струн в електричний сигнал, який може бути надалі посилений або

записаний. Із сфери використання можна судити про точність такого датчика, тому п'єзоелектричний звукознімач підійде, для проектів де точність та відсутність шумів надважлива. Існує велика кількість п'єзоелемент різних конфігурацій, штучних та природних, які мають різні властивості, тому є можливість підібрати необхідний для конкретної задачі [5].

3. Опис досліджень

Для наглядного порівняння усіх трьох датчиків SW-520D, MPU-6050 та п'єзоелектричного звукознімача було створено схему на базу Ардуіно, щоб усі 3 датчики зчитували інформацію одночасно, було проведено декілька вимірювань з різною силою, амплітудою та інтенсивність коливань. На основі отриманих даних, було створено порівняльні графіки на яких можна побачити як під час експерименту ведуть себе різні пристрої. Окремий дослід був проведений у стані відносного покою, для виміру шуму.

Схема зібрана на основі Arduino Uno [6] мікроконтролера на базі ATmega328P [7]. Він має 14 цифрових контактів входу/виходу (з яких 6 можна використовувати як ШІМ виходи), 6 аналогових входів, керамічний резонатор 16 МГц (CSTCE16M0V53-R0), USB-з'єднання, роз'єм живлення, роз'єм ICSP. Схема відносно проста, складається з самої плати Arduino Uno, датчика SW-520D, резистора на 10 кОм, датчика MPU-6050 та п'єзоелектричного датчика Avzhezh P-007, який було розроблено для музичних інструментів. Усі датчики окрім п'єзоелектричного надсилають сигнал до Ардуіно, а п'єзоелектричний звукознімач напряму до комп'ютера як мікрофон.

Оскільки датчики працюють у різному діапазоні, результати необхідно опрацювати для наглядності порівняння, таким чином результат від цифрового датчика SW-520d тепер показує 100 при виявленні коливань, а данні з MPU-6050 поділені на 20000 та помножені на 100.

3.1. Опис схеми

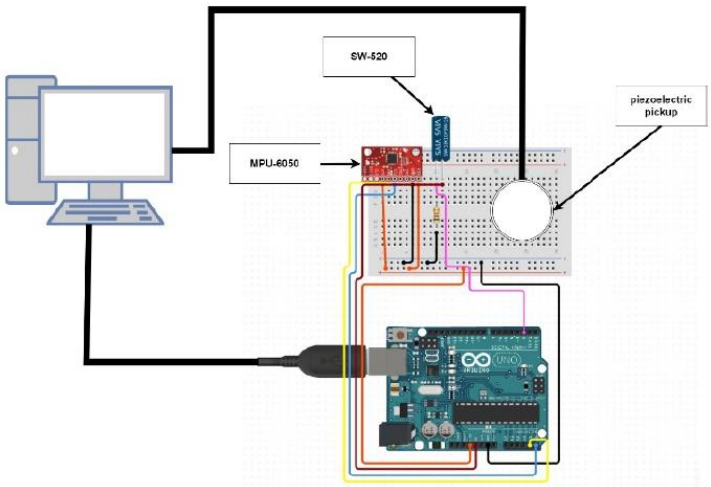


Рисунок 3 - Схема приладу

Датчик Sw-520D підключений до третього цифрового виходу, коли кульки у середині сенсора замикають ланцюг цифровий вихід номер три, на платі,

починає отримувати одиниці, тобто напругу у розмірі 3,3 вата. Сенсор MPU-6050 має більш складний інтерфейс підключення, вихід VCC підключений до плюса, GND до землі, виходи SCL та SOA підключаємо до аналогових виходів A5, A4 відповідно. Виходи A4 та A5 обумовлені бібліотекою MPU6050.h, яку було використано при написанні коду [9]. П'єзоелектричний звукознімач підключений до комп'ютера напряму через Jack (1/4") - Mini-Jack 1/8" (3.5 мм) перехідник. Уся схема працює від джерела енергії 3,3 вата, та розташована на макетній платі [7].

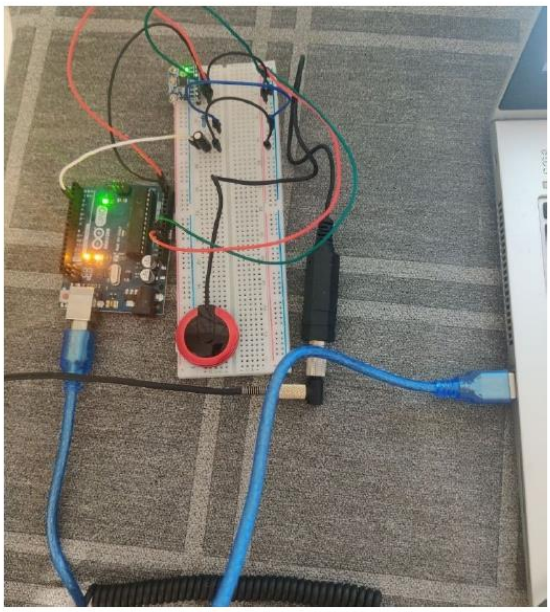


Рисунок 4 - Вигляд пристрою

3.2. Опис результатів дослідження #1

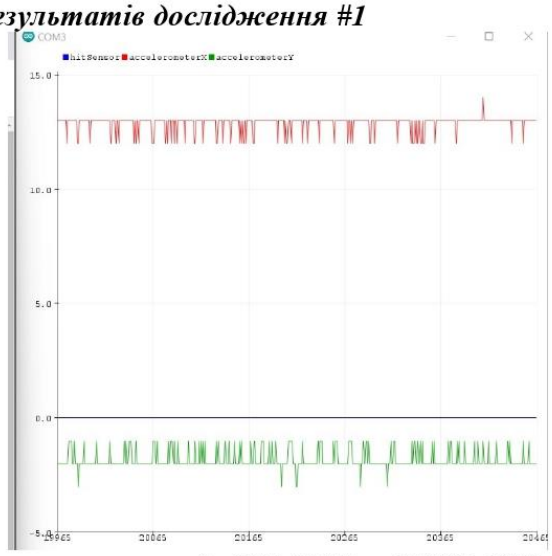


Рисунок 4 - Результати з датчиків SW-520d та MPU-6050 для першого тесту

Перше дослідження було проведено у стані відносного спокою, на пристрій, та на поверхню на якій він знаходиться не впливали механічно, тому результати можна розцінювати як побічний шум, який виникає при роботі різних датчиків.

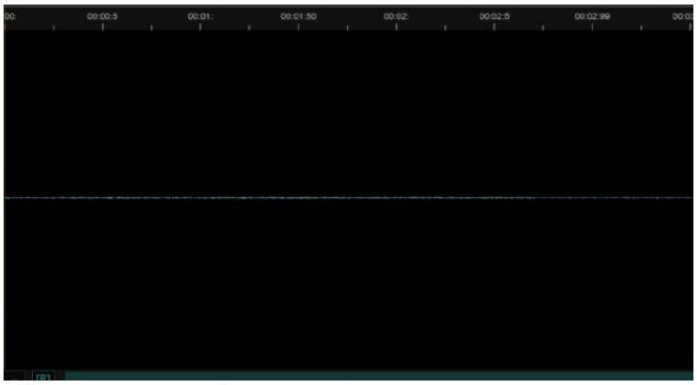


Рисунок 5 - Результати з п'єзоелектричного датчика для першого тесту

У результаті, на графіку, видно, що найбільша амплітуда коливань спостерігається у акселерометра для датчика MPU-6050, амплітуда коливань по осі X та Y дорівнює 5 умовним позначкам. Показник датчика SW-520D, синій лінія на графіку, незмінний протягом усього тесту, а п'єзоелектричний датчик реєструє малі коливання, мабуть від роботи куллера ноутбуку [8].

3.3. Опис результатів дослідження #2

У другому досліді поверхня на якій знаходиться пристрій, у даному випадку це стіл, було під механічним впливом, приблизно на відстані 10 сантиметрів до макетної плати, на якій знаходяться усі датчики. Ритмічні удари відбувалися раз у секунду під час усього експерименту. На графіках зображено однаковий період часу для усіх датчиків.

Рисунок 6 - Результати з датчиків SW-520d та MPU-6050 для другого тесту

Рисунок 7 - Результати з п'єзоелектричного датчика для другого тесту

Як можемо побачити на графіках сили удару по поверхі не достатньо для роботи сенсора SW-520D, синя лінія на фігурі шість незмінна, навіть при третьому ударі, якій відносно сильніший ніж інші п’ять. Датчик MPU-6050, точно зареєстрував усі шість ударів по поверхні і заміряв їхню силу, амплітуда коливань сягала двадцяти п'яти одиницям. Комбінація показників акселерометра по осі X та Y, може збільшити загальну точність пристрою, якщо коливання відбуваються не у одному напрямку, чи сам пристрій знаходиться під нахилом. П'єзоелектричний датчик не зміг зареєструвати усі шість ударів, а тільки три найпотужніші, які не дуже чітко видно на графіку. Показники п'єзоелектричного датчика збільшили шуми, що може вказувати на його здатність реєструвати не тільки сильні удари, а й затухаючі коливання, які виникають після.

3.4. Опис результатів дослідження #3

У третьому досліді легкі удари приходилися уже по самому пристрою, а саме по вільному простір на макетній платі між усіма датчиками. Так само, як і у другому тесті, ритмічні удари відбувалися раз у секунду під час усього експерименту. На графіках зображено однаковий період часу для усіх датчиків.

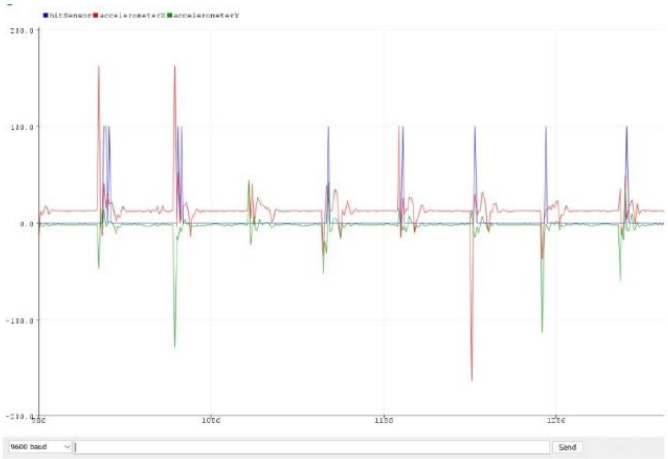


Рисунок 8 - Результати з датчиків SW-520d та MPU-6050 для третього тесту

Рисунок 9 - Результати з датчиків SW-520d та MPU-6050 для третього тесту

Під час третього тесту датчик SW-520D почав реєструвати коливання, синя лінія на графіку, це очікувано, тому що такий датчик позначено як сенсор удару, тому такий результат вважаю задовільним, хоч він і не показав один із максимумів, які зареєстрували інші сенсори.

Акселерометр датчика MPU-6050, у порівнянні з другим тестом тільки збільшив амплітуду, тому що сила удару збільшилися, так само точно зареєстрував усі вісім ударів по поверхні, та відобразив їх на графіку з показником сили для кожного.

П'єзоелектричний датчик показав себе найкраще, у третьому досліді, на графіку можна побачити вісім максимумів, та те як матеріал пристрою коливається після ударів.

Висновоки

У результаті дослідження було проведено три тести з трьома різними сенсорами, які використовуються у різних сферах, але мають спільну здібність до реєстрації коливань на поверхні, під час дослідження особлива увага приділялася точності сенсорів у різних умовах, таким чином можна зробити висновки окремо для кожного датчика.

Сенсор SW-520D, має найменшою чутливістю серед представлених, його застосовують, як датчик нахилу або удару, тому лише при прямих ударах по поверхні пристрою він їх реєструє, такий сенсор може бути корисним для автомобілів, або у сферах де відбуваються високоамплітудні коливання, а бо удари і не важлива сила удару, тільки його наявність.

MPU-6050 показав себе найкраще серед представлених, акселерометр, хоч і показує найбільший рівень шумів, але зміг точно розпізнати усі удари як по корпусу пристрою, так і по поверхні на який він знаходиться. Через наявність гіроскопу, акселерометра та термометра, кількість сфер та проектів де така плата буде корисна, дуже велика.

П'єзоелектричний датчик найчутливіший з представлених, його графік надає найбільше інформації про коливання, але його робота залежить від

підсилювачів, як і будь якого пристрою який працює зі звуком. Пристрій Avzhezh P-007 являє собою мікрофон для корпусу гітар і повинен використовуватися у комбінації з підсилювачем, але данні, які можна отримати від такого датчика найбільш точно показують малоаплітудні коливання такі як звук.

У висновку хочу додати, що кожен з сенсорів потрібно підбирати виходячи з задачі, а універсального датчика не існує.

Література:

[1] SW-520D Documentation [Електронний ресурс] – https://www.tme.com/Document/f1e6cedd8cb7feeb250b353b6213ec6c/SW-520D.pdf.

[2] MPU-6050 Datasheet [Електронний ресурс] – https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf

[3] Padma Nyoman Crisnapati, Putu Desiana Wulaning, I Nyoman Rudy Hendrawan, Anak Agung Ketut Bagus Bandanagara, "Earthquake Damage Intensity Scaling System based on Raspberry Pi and Arduino Uno," The 6th International Conference on Cyber and IT Service Management (CITSM 2018), Inna Parapat Hotel – Medan, August 7-9, 2018

[4] N. K. Wargantiwar, A. S. Barbade, A. P. Shingade, A. N. Shire, "Wireless Earthquake Alarm Design based on MEMS Accelerometer," International Advanced Research Journal in Science, Engineering and Technology, vol. 4, Special Issue 3, pp. 128-132, January 2017

[5] П'єзоелектричний звукознімач [Електронний ресурс] - https://uk.wikipedia.org/wiki/Електромагнітний_звукознімач.

[6] Picking the Right Arduino [Електронний ресурс] - Режим доступу до ресурсу: https://medium.com/@baldengineer/picking-the-right-arduino-341a0a9550c7

[7] Arduino ATmega328P Datasheets Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://content.arduino.cc/assets/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

[8] Richard J. Smythe "Arduino Measurements in Science, Advanced Techniques and Data Projects".

Abstract. *Research on the registration of vibrations on the surface using a computer hardware platform for amateur design Arduino reveals modern ways to create a physical product to work with different types of surface oscillations. The system has many sensors compatible with the platform, and components that can be individually adapted to work, the microcontroller board with I/O elements and the Processing/Wiring development environment for Arduino. The task was to conduct three experiments with three different sensors in different conditions, and to investigate the ability of Arduino to adapt to different tasks and environments. During the study, three tests were performed to collect data on the operation of sensors that were in the same conditions. The purpose of this work is to identify ideal conditions and determine the technical capabilities of each of the sensors, for which the conditions in each test were different, as well as proving the feasibility of using the Arduino system to create prototypes. The results of the study show that there is no universal sensor for measuring surface vibrations, and each of the presented sensors is suitable for individual amplitude ranges and specific purposes. The study concludes that when creating a*

product that has the ability or need to measure oscillations on the surface, it is necessary to understand what oscillations the device will work with, and this plays a key role in choosing the right sensor for the Arduino system..

Key words: *Arduino, fluctuation, vibration sensor.*

Стаття відправлена: 19.04.2022 г.