НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ

Завідувач кафедри
Комп'ютерних систем, мереж та кібербезпеки
_____ Касаткін Д.Ю., к.пед.н., доц.
(підпис) (ПІБ, вчене звання і ступінь)

«___»_____2025 р.

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

На тему: «Розробка бездротової автономної системи вимірювання параметрів руку об'єкту на основі технології Bluetooth»

Спеціальність 123 «Комп'ютерна інженерія»

/ Нікітенко Є.В.

Гарант освітньої програми
к.фіз.-мат.н., доц.
(підпис) (ПІБ)

Керівник дипломного проекту: _____ / Сагун А.В. /
(підпис) (ПІБ)

Виконав: _____ / Процик М.С. /
(підпис) (ПІБ)

КИЇВ-2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

«ЗАТВЕРДЖУЮ»
завідувач кафедри
комп'ютерних систем, мереж та кібербезпеки
_____/ Касаткін Д.Ю., к.пед.н., доц. /
(підпис) (ПІБ, вчене звання і ступінь)
«__» _____ 20__ р.

З А В Д А Н Н Я

ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ БАКАЛАВРСЬКОЇ СТУДЕНТУ

Процика Максима Сергійовича
(прізвище, ім'я, по батькові)
Спеціальність (напрям підготовки): комп'ютерна інженерія
Тема кваліфікаційної бакалаврської роботи: «Розробка бездротової автономної системи вимірювання параметрів руку об'єкту на основі технології Bluetooth
затверджена наказом ректора НУБіП України від "16" 12 2024р. № 2248 "С"
Термін подання завершеної роботи на кафедру _____
Вихідні дані до кваліфікаційної бакалаврської роботи: електросхеми пристрою, значення напруги, схемні ілюстрації
Перелік питань, що підлягають розробці:
1. Аналіз існуючих систем контролю мікроклімату в приміщенні
2. Вибір мікроконтролерів та датчиків для розробки системи контролю температури
3. Розробка програмного забезпечення

Перелік графічного матеріалу (за потреби) _____

Дата видачі завдання "_____" _____ 2025 р.

Керівник кваліфікаційної роботи _____ Сагун А.В., к.т.н.
(підпис) (прізвище та ініціали)

Завдання прийняв до виконання _____ _____
(підпис) (прізвище та ініціали студента)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання бакалаврської кваліфікаційної роботи	Строк виконання етапів бакалаврської кваліфікаційної роботи	Примітка
1	Отримання завдання бакалаврської кваліфікаційної роботи	16.12.2024	
2	Початок планування системи	23.01.2025 – 03.02.2025	
3	Побудова UML діаграм	25.03.2025-27.04.2025	
4	Розробка програми і тестування	08.03.2025-23.04.2025	
5	Написання пояснювальної записки	24.04.2025-16.05.2025	
6	Перевірка на плагіат	17.05.2025	
7	Відправка дипломної записки	17.05.2025	
8	Захист дипломної роботи	25.05.2025-15.06.2025	

Студент _____ _____Процик М.С._____

(підпис) (ПІБ)

Керівник бакалаврської кваліфікаційної роботи

_____ _____Сагун А.В._____

(підпис) (ПІБ)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

– BLE (Bluetooth Low Energy) – протокол бездротового зв’язку з низьким енергоспоживанням, який використовується для передавання даних між мікроконтролером і зовнішніми пристроями.

– IMU (Inertial Measurement Unit) – інерціальна вимірювальна система, що складається з акселерометра, гіроскопа, а іноді й магнітометра, для реєстрації руху та орієнтації.

– EMA (Exponential Moving Average) – експоненційне згладжування, алгоритм цифрової фільтрації сигналів для усунення шумів і підвищення стабільності даних.

– ADC (Analog-to-Digital Converter) – аналогово-цифровий перетворювач, який дозволяє зчитувати аналогові сигнали з датчиків і конвертувати їх у цифрові значення.

– PWM (Pulse Width Modulation) – широтно-імпульсна модуляція, метод керування сервоприводами за допомогою зміни тривалості імпульсу.

– Vref (Voltage Reference) – опорна напруга, що використовується АЦП для визначення діапазону вимірювання аналогових сигналів.

– Peripheral Mode – режим роботи пристрою BLE, у якому він виступає периферійним модулем і передає дані на центральний пристрій.

– Notify Mode – режим передачі повідомлень у BLE, за якого периферійний пристрій ініціює надсилання даних до центрального.

– Arduino IDE (Integrated Development Environment) – інтегроване середовище розробки для написання, компіляції та завантаження коду у мікроконтролери Arduino.

– PlatformIO – розширене середовище для розробки прошивок для мікроконтролерів з підтримкою структурованих проєктів і менеджера бібліотек.

– G-force (g) – одиниця прискорення, еквівалентна прискоренню вільного падіння (~9.81 м/с²), що використовується для оцінки лінійного прискорення.

– I²C (Inter-Integrated Circuit) – інтерфейс синхронного послідовного зв’язку між мікросхемами (не використовується в поточному проєкті, але часто застосовується у сенсорах IMU).

– RX/TX – канали прийому (receive) та передачі (transmit) даних послідовного інтерфейсу (UART).

– Serial Monitor – інструмент Arduino IDE для перегляду даних, що надходять через UART у режимі реального часу.

– dt (delta time) – зміна часу між двома вимірами, яка використовується для обчислення інтегрованого значення кутової швидкості.

– mAh (milliampere-hour) – міліампер-година, одиниця вимірювання ємності акумулятора або енергоспоживання.

– OLED (Organic Light-Emitting Diode) – тип дисплея, який іноді використовується для виводу результатів у подібних системах (у цьому проєкті не використовується, але можливий при масштабуванні).

– LoRa (Long Range) – протокол бездротової передачі даних на великі відстані з низьким енергоспоживанням (можливий напрямок масштабування).

ВСТУП

Сучасні тенденції в автоматизації та цифровізації обумовлюють зростаючу потребу в автономних системах збору та аналізу параметрів руху об’єктів. Такі системи активно застосовуються у сферах біомедичних досліджень, спортивної науки, робототехніки, контролю логістичних процесів, а також у промислових і побутових розробках [14; 15]. Їх ефективність значною мірою залежить від здатності точно вимірювати просторові характеристики об’єкта, працювати в автономному режимі та безперебійно передавати інформацію на зовнішні пристрої для подальшої обробки.

Один з найперспективніших напрямів побудови таких систем — використання бездротових інерційних модулів із підтримкою Bluetooth Low Energy (BLE), які поєднують низьке енергоспоживання, компактність і здатність передавати дані в реальному часі [13; 21]. Застосування мікроконтролерів типу ESP32, що інтегрують Bluetooth та Wi-Fi, дозволяє створювати багатофункціональні пристрої з відкритою архітектурою, адаптовані до широкого спектру завдань [22]. Інерційні сенсори, зокрема MPU-6050, забезпечують вимірювання прискорення та кутової швидкості, що дозволяє здійснювати комплексний аналіз динаміки руху.

Однак практичне впровадження таких систем стикається з низкою проблем, серед яких: обмеженість енергетичних ресурсів, складність обробки сигналів сенсорів в умовах шуму, нестабільність каналу передачі даних, а також відсутність адаптивних архітектур для інтеграції в реальні прикладні сценарії [5; 6]. Більшість комерційних рішень орієнтовані на закриті платформи, що ускладнює адаптацію або модифікацію під нестандартні вимоги.

Метою даної роботи є розробка бездротової автономної системи вимірювання параметрів руху об’єкта, яка базується на мікроконтролері з Bluetooth-модулем та інерційним сенсором, забезпечує автономну обробку,

зберігання та передачу даних у режимі реального часу, з можливістю подальшої масштабованості та застосування у різних доменах.

Для досягнення поставленої мети необхідно вирішити такі **завдання:**

1. Проаналізувати існуючі технології побудови автономних бездротових систем вимірювання параметрів руху;
2. Обґрунтувати вибір апаратної платформи та сенсорів;
3. Розробити структурну архітектуру системи та визначити її функціональні компоненти;
4. Реалізувати алгоритми збору, обробки та передачі даних з урахуванням обмежень по енергоспоживанню;
5. Створити програмне забезпечення для взаємодії з мобільними або ПК-пристроями через Bluetooth;
6. Провести тестування працездатності, точності вимірювань та стабільності передачі даних;

Об’єктом дослідження є автономна сенсорна система збору та передачі даних про параметри руху об’єкта.

Предметом дослідження є апаратно-програмні рішення з використанням інерційних сенсорів та Bluetooth для побудови систем автономного моніторингу.

Практична значущість дослідження полягає у створенні універсального технічного рішення з відкритою архітектурою, придатного для персонального, освітнього або дослідницького застосування в умовах обмежених ресурсів. Розроблена система може бути інтегрована у більші комплексні рішення (наприклад, IoT або системи керування), що забезпечує її масштабованість і технологічну гнучкість [14; 24].

Структура кваліфікаційної роботи включає вступ, три розділи, висновки, список використаних джерел та додатки. У першому розділі проведено аналітичний огляд предметної області, досліджено сучасні технічні рішення . У другому розділі виконано проєктування апаратної та програмної частини

системи, визначено основні алгоритми функціонування. У третьому розділі здійснено реалізацію прототипу, його тестування та аналіз результатів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Актуальність дослідження автономних систем вимірювання руху

У сучасних умовах інтенсивного розвитку цифрових технологій, зокрема в галузях мобільної робототехніки, біомедичних систем, спорту, логістики та «розумного» середовища, спостерігається зростаюча потреба в ефективних, компактних і автономних засобах вимірювання параметрів руху. Системи такого типу повинні забезпечувати високоточне вимірювання прискорення, кутових швидкостей, положення або переміщення об'єкта з мінімальним втручанням у його природну динаміку. Це особливо важливо для мобільних, носимих або вбудованих застосувань, де кабельне підключення є неприйнятним з точки зору мобільності, енергетичної ефективності та ергономіки [13].

Однією з критичних проблем класичних дротових систем збору даних є їхня фізична обмеженість, яка проявляється в низькій адаптивності до мобільних умов, складності розгортання, а також високому рівні енергоспоживання через постійну потребу в живленні та передачі даних через кабель [8]. Крім того, такі системи часто орієнтовані на стаціонарні або лабораторні сценарії використання, що істотно знижує їх застосовність у реальних умовах — зокрема у спорті, польових вимірюваннях або персональному моніторингу [3].

З розвитком технологій низькоенергетичного бездротового зв’язку, зокрема Bluetooth Low Energy (BLE), стало можливим створення нових архітектур автономних сенсорних систем, які мають мінімальні вимоги до джерел живлення, здатні функціонувати протягом тривалого часу від компактних акумуляторів і водночас забезпечувати стійку передачу даних на

зовнішні приймачі — смартфони, планшети, комп’ютери або мікроконтролерні шлюзи [27]. BLE-модулі, інтегровані у мікроконтролери, як-от ESP32, мають вбудовані комунікаційні стеки, периферійні інтерфейси та підтримку роботи в режимі глибокого сну, що суттєво розширює можливості автономної роботи [22].

Іншим важливим компонентом таких систем є інерційні сенсори, зокрема MPU-6050, які забезпечують одночасне вимірювання лінійного прискорення та кутових швидкостей по трьох осях. Ці сенсори працюють на базі MEMS-технології (microelectromechanical systems) і поєднують в одному корпусі акселерометр та гіроскоп, що робить їх придатними для вимірювальних задач у системах з обмеженим об’ємом і вагою [20].

На рисунку 1.1 наведено логіко-структурну схему, яка ілюструє зв'язок між трьома ключовими аспектами: проблемами традиційних дротових підходів, потребами сучасних галузей та перевагами автономних бездротових рішень.

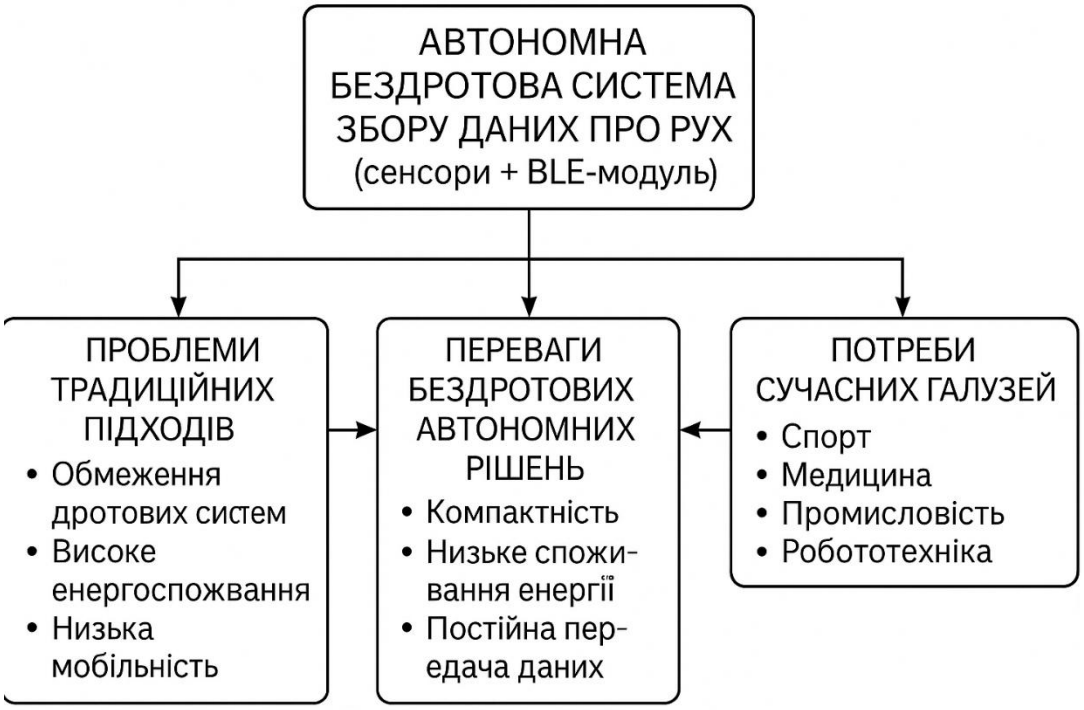

Рисунок 1.1 – Актуальність впровадження автономних систем збору параметрів руху

Центральним елементом виступає бездротова автономна система збору руху, побудована на базі сенсорів та BLE-модуля, яка акумулює переваги технологічної інтеграції та покликана задовольнити актуальні прикладні запити.

У межах концепції Інтернету речей (IoT) такі автономні модулі виконують роль інтелектуальних периферійних вузлів, здатних збирати інформацію, попередньо її обробляти і передавати до центральних систем без участі людини. Унаслідок цього підвищується ефективність аналізу, швидкість реакції на зміни в системі та зменшуються витрати на обслуговування [14]. Водночас системи стають гнучкими до змін умов роботи, що критично важливо в контексті динамічних середовищ — таких як спортивні змагання, мобільні обстеження пацієнтів або маневрування автономних роботів [5].

У таблиці 1.1 наведено технічне порівняння традиційних дротових і сучасних автономних бездротових систем збору параметрів руху за критичними критеріями, що визначають їх придатність до використання у прикладних умовах.

Таблиця 1.1 – Порівняння традиційних та автономних систем вимірювання руху

Критерій	Дротові системи	Автономні бездротові системи (BLE)
Тип живлення	Постійне живлення по кабелю	Вбудоване автономне джерело
Мобільність	Низька	Висока
Тип передачі	Кабель	Бездротова (Bluetooth BLE)
Точність сенсорики	Висока	Висока
Енергоспоживання	Високе	Низьке
Масштабованість системи	Обмежена	Гнучка
Інтеграція з мобільними пристроями	Ускладнена	Повноцінна підтримка BLE

Узагальнюючи варто відзначити , що впровадження автономних бездротових систем збору параметрів руху є обґрунтованим з технічної, економічної та прикладної точок зору. Такі системи здатні задовольнити вимоги

до енергоефективності, мобільності, точності та масштабованості, що робить їх ключовим елементом у розвитку персоніфікованих цифрових технологій та автономних інформаційно-керуючих структур нового покоління.

1.2 Огляд технологій бездротової передачі даних: порівняння Bluetooth, Wi-Fi, ZigBee

Бездротовий обмін даними є фундаментальною складовою сучасних телекомунікаційних і вбудованих систем. Його суть полягає у передаванні цифрової інформації між пристроями без фізичного провідника шляхом модуляції електромагнітних хвиль у певному частотному діапазоні. Найбільш поширеними є неліцензовані ISM-діапазони, зокрема 2.4 ГГц, 5 ГГц, а також вузькополосні спектри нижче 1 ГГц, які забезпечують баланс між дальністю дії та швидкістю передачі [14].

Бездротова система зв’язку, незалежно від конкретного стандарту (Bluetooth, Wi-Fi, ZigBee), базується на передавачі, приймачі, антенному блоці, цифровому протоколі та енергетичному інтерфейсі. Особливої актуальності це набуває у сенсорних пристроях із автономним живленням, де критично важливо забезпечити оптимізацію енергоспоживання, надійність зв’язку та компактність реалізації [5].

У контексті побудови мікроконтролерних систем вимірювання руху доцільно розглядати такі технічні параметри, які безпосередньо впливають на придатність бездротового стандарту до використання в автономних мобільних рішеннях. Ці параметри узагальнено у таблиці 1.2.

Таблиця 1.2 – Основні параметри для порівняння технологій бездротового зв’язку

Параметр	Опис
Енергоспоживання	Визначає витрати енергії при активному, пасивному та сплячому режимах роботи
Швидкість передачі даних	Пропускна здатність каналу; визначає обсяг інформації, яку можна передати за одиницю часу

Дальність дії	Максимальна ефективна відстань між передавачем і приймачем у типовому середовищі
Топологія мережі	Тип побудови мережі: точка-точка, зірка, дерево, mesh – впливає на гнучкість і надійність

Продовження таблиці 1.2

Масштабованість	Можливість збільшення кількості вузлів у мережі без втрати якості обслуговування
Час встановлення зв’язку	Затримка при ініціалізації з’єднання між пристроями, важлива для динамічних систем
Інтерфейси підключення	Підтримка UART, I²C, SPI, GPIO для взаємодії з мікроконтролером або системою на чипі
Стійкість до перешкод	Здатність зберігати якість зв’язку при наявності радіошуму або перешкод у каналі
Сумісність	Можливість інтеграції з існуючими апаратними платформами та мобільними пристроями

Згідно з [20] та [22], для автономних сенсорних систем вирішальними є показники енергоспоживання, часу виходу з режиму сну, стабільності передачі даних у реальному часі, а також наявність підтримки стандартних інтерфейсів мікроконтролера. Ці характеристики дозволяють забезпечити баланс між енергоефективністю та функціональністю у портативних і мобільних додатках.

Bluetooth Low Energy (BLE) є сучасною специфікацією стандарту Bluetooth, оптимізованою для енергозберігаючого обміну невеликими обсягами даних між пристроями, що живляться автономно. Цей стандарт реалізує комунікацію в частотному діапазоні 2,4 ГГц, використовуючи технологію розширення спектру та адаптивне перемикання частот для зниження перешкод і збільшення надійності зв’язку. BLE функціонує за архітектурною моделлю "Central–Peripheral", де пристрій Central (наприклад, смартфон) ініціює з'єднання з периферійним вузлом, яким виступає мікроконтролер із сенсором [22].

Протокольна модель BLE включає кілька рівнів: Link Layer, L2CAP, Attribute Protocol (ATT) та Generic Attribute Profile (GATT), які забезпечують повноцінну логіку передачі даних, збереження їхньої структури та узгоджене керування обміном. Саме GATT відповідає за логічне структурування сервісів і

характеристик, які можуть бути зчитані або записані з використанням стандартних UUID-ідентифікаторів. Цю ієрархію показано на рисунку 1.2, де видно основні програмно-апаратні рівні та логіку комунікації між пристроями.

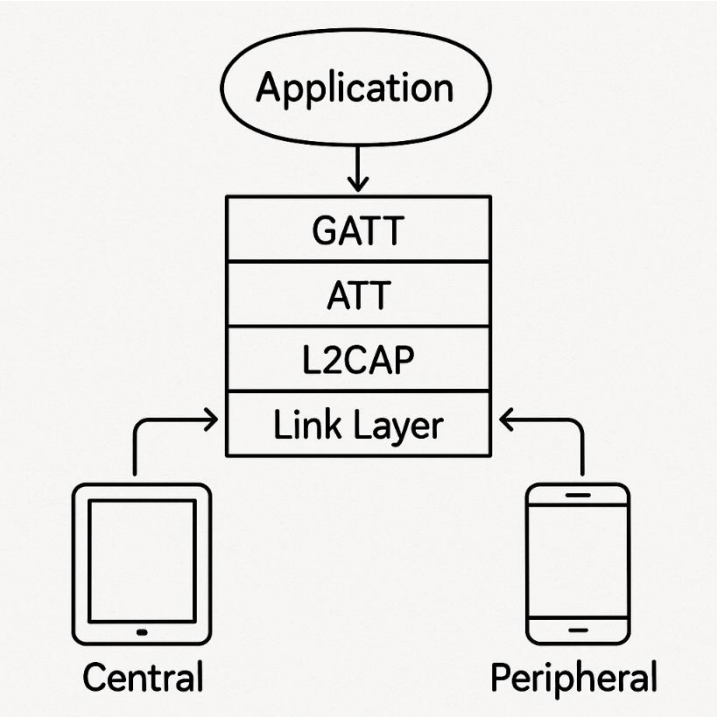

Рисунок 1.2 – Протокольна структура BLE та архітектура з'єднання Central–Peripheral

BLE має кілька вагомих переваг, що роблять його придатним до застосування в автономних системах: наднизьке енергоспоживання (у режимі передачі ~0.01–0.05 мА·год), підтримка роботи з мобільними платформами без потреби в додатковому устаткуванні, проста реалізація у мікроконтролерах типу ESP32, наявність широкої бази відкритих бібліотек і прикладів [27]. Обмеженням BLE виступає відносно мала пропускна здатність (до 1 Мбіт/с) та обмежена кількість одночасних з’єднань у ролі Peripheral (типово до 4–5 пристроїв залежно від реалізації).

На відміну від BLE, технологія Wi-Fi (IEEE 802.11x) орієнтована на високошвидкісну передачу великих обсягів інформації та здебільшого використовується в інфраструктурних мережах за схемою "точка доступу – клієнт". У типових умовах Wi-Fi забезпечує швидкість від 11 до 1000 Мбіт/с,

залежно від версії стандарту (802.11b/g/n/ac), і потребує значно більше енергії (~50–300 мА в активному режимі) [13]. Це робить його малопридатним для сенсорних систем, де необхідна автономність на рівні днів або тижнів. До того ж, Wi-Fi має тривалий час пробудження з режиму сну, що ускладнює його використання в подієво-орієнтованих застосуваннях із нерегулярним обміном. Загальні принципи Wi-Fi-з'єднання проілюстровано на рисунку 1.3, де також акцентовано ключові сценарії використання та недоліки в контексті автономних рішень.

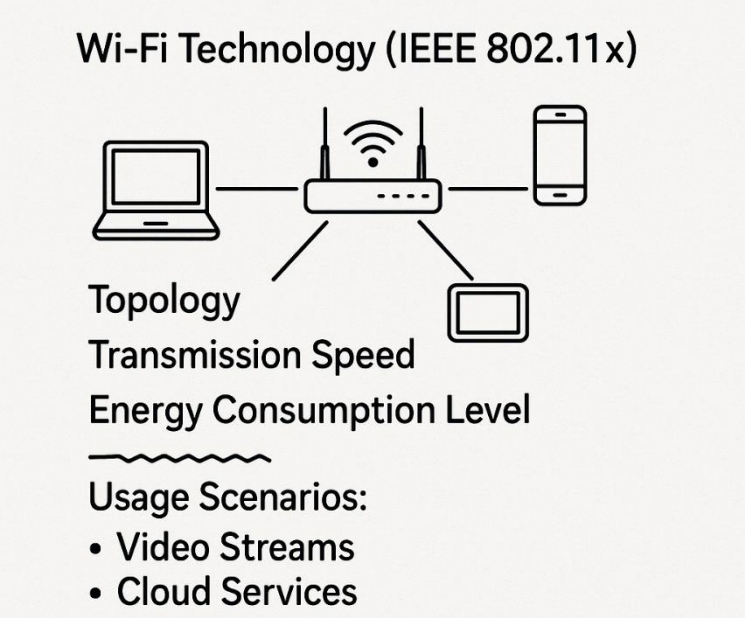

Рисунок 1.3 – Особливості архітектури Wi-Fi у сенсорних застосуваннях

Ще одним популярним стандартом є ZigBee (IEEE 802.15.4), який функціонує на тій же частоті, що й BLE (2.4 ГГц), але підтримує сітчасту (mesh) топологію з розширеною маршрутизацією, що дозволяє об'єднувати десятки й сотні вузлів у стійкі мережі з автоматичним перепрокладанням маршрутів. ZigBee споживає мало енергії (менше ніж Wi-Fi, але більше ніж BLE) та підтримує швидкість передачі до 250 Кбіт/с. Стандарт здатний забезпечувати довготривалу роботу сенсорів у масштабованих мережах, що особливо цінно в індустріальних або екологічних системах моніторингу [20].

Водночас, ZigBee має певні обмеження щодо інтеграції з мобільними пристроями, оскільки для взаємодії з ним потрібні спеціалізовані координатори

або шлюзи. Також конфігурація і підтримка сітчастої мережі вимагає вищої складності програмної логіки та протоколів безпеки [5]. Архітектура ZigBee загалом менш орієнтована на портативні індивідуальні пристрої, що робить її менш привабливою в контексті персоналізованих рішень.

Після детального аналізу кожної з розглянутих технологій доцільно провести їх системне порівняння за низкою ключових технічних параметрів, що безпосередньо впливають на ефективність впровадження у складі автономної сенсорної системи вимірювання параметрів руху. Основними критеріями для порівняння виступають швидкість передачі даних, максимальна дальність дії, рівень енергоспоживання в активному та пасивному режимах, масштабованість архітектури мережі, а також ступінь сумісності із популярними мікроконтролерами, зокрема такими, як ESP32 або STM32. Для забезпечення об'єктивного зіставлення ці параметри зведено у таблицю 1.3.

Таблиця 1.3 – Порівняння технологій BLE, Wi-Fi та ZigBee за критичними параметрами

Параметр	Bluetooth (BLE)	Wi-Fi (802.11n)	ZigBee (802.15.4)
Швидкість передачі	до 1 Мбіт/с	до 150 Мбіт/с	до 250 Кбіт/с
Дальність дії	10–30 м (до 100 м в LoS)	30–100 м (залежно від AP)	10–100 м (в мережі >1 км)
Енергоспоживання	дуже низьке	високе	низьке
Масштабованість	до 20 з’єднань	обмежена мережею	>100 пристроїв (mesh)
Сумісність з мікроконтролерами	висока (вбудовані BLE)	помірна (Wi-Fi-модулі)	потребує координаційного шлюзу

Як показує порівняльний аналіз, Bluetooth Low Energy забезпечує оптимальний баланс між ключовими вимогами до енергоефективності, габаритів, швидкості та простої реалізації в мобільних або портативних системах. Висока сумісність з сучасними мікроконтролерними платформами, наявність вбудованих BLE-модулів, таких як ESP32, а також підтримка

стандартних інтерфейсів зв’язку (UART, I²C, SPI) забезпечують легкість інтеграції в архітектуру вимірювального пристрою. Окрім того, специфікація BLE підтримує гнучкі механізми управління живленням і має швидкий час пробудження з режимів сну, що критично важливо для періодичних або подієво-орієнтованих задач збору даних.

У порівнянні з Wi-Fi, BLE не тільки споживає на порядок менше енергії, але й не потребує постійного підключення до точки доступу, що усуває додаткові вимоги до інфраструктури. У свою чергу, ZigBee, попри високу масштабованість і підтримку mesh-топології, потребує складнішої організації мережі та наявності координатора, що суперечить принципу мінімізації апаратної складності у компактних автономних системах. Додатковим обмеженням ZigBee є його обмежена сумісність із мобільними платформами без спеціалізованих шлюзів, що ускладнює реалізацію збирання даних у персоналізованих або споживчих рішеннях.

Зважаючи на усі технічні та експлуатаційні аспекти, Bluetooth Low Energy є найбільш доцільним вибором для побудови бездротової автономної системи збору параметрів руху об'єкта. Його використання забезпечує ефективний компроміс між точністю, тривалістю автономної роботи, простотою розробки і масштабованістю для подальшої інтеграції в більш складні IoT-системи або платформи моніторингу.

1.3 Аналіз існуючих систем моніторингу руху об’єктів

У сучасній практиці збору кінематичних параметрів об'єктів, таких як прискорення, швидкість, кутова орієнтація та положення у просторі, активно використовуються вбудовані бездротові системи моніторингу руху. Такі системи знаходять застосування в біомеханіці, спорті, медицині, віртуальній та доповненій реальностях, а також у промислових та робототехнічних рішеннях.

Їх основною функцією є автономне вимірювання динаміки об’єкта у реальному часі з передачею даних по бездротових каналах, найчастіше за допомогою Bluetooth або Wi-Fi. Нижче розглянуто низку відомих комерційних систем, що відзначаються технічною довершеністю та високою точністю вимірювань.

Одним з поширених рішень є Movesense — компактний інерційний сенсор, розроблений компанією Suunto. Пристрій має форму диска з магнітним кріпленням і поєднує тривісний акселерометр, гіроскоп та магнітометр. Він підтримує передачу даних через Bluetooth Low Energy, що забезпечує низьке енергоспоживання та високу сумісність з мобільними платформами. Завдяки відкритому SDK, Movesense широко використовується в експериментальних системах аналізу руху спортсменів, в реабілітаційній медицині та фітнес-пристроях. Його компактність і герметичність дозволяють розміщувати сенсор безпосередньо на тілі або спортивному спорядженні. Зовнішній вигляд та форм-фактор пристрою наведено на рисунку 1.4.

Рисунок 1.4 – Компактний інерційний сенсор Movesense із Bluetooth-зв'язком

Ще одним потужним рішенням є Xsens DOT — професійний модуль для моніторингу руху, орієнтований на високоточні задачі. Він оснащений інерційним вимірювальним блоком (IMU) з гіроскопом, акселерометром та магнітометром, об'єднаними апаратною фільтрацією і алгоритмами трекінгу положення. Xsens підтримує передачу даних по Bluetooth 5.0, має вбудований буфер пам’яті для автономної роботи та забезпечує частоту зчитування до 120

Гц. Завдяки використанню алгоритмів фільтрації шумів та компенсації дрейфу, система забезпечує високу точність у просторовому позиціюванні. Xsens активно застосовується у спортивній біомеханіці, клінічних дослідженнях рухової активності, а також у виробництві анімації для цифрових середовищ. Зовнішній вигляд пристрою ідентифікується на рисунку 1.5.

Рисунок 1.5 – Сенсорний модуль Xsens для високоточного трекінгу руху

Ще однією перспективною платформою у класі автономних носимих сенсорів є MetaMotionR, розроблена компанією MbientLab. Пристрій виконаний у вигляді мініатюрного модуля з захисним силіконовим кейсом для фіксації на тілі або об'єкті. Він оснащений тривісними акселерометром, гіроскопом і магнітометром, підтримує Bluetooth Low Energy і має вбудовану флеш-пам’ять для локального зберігання даних. Однією з переваг MetaMotionR є підтримка SDK для мобільних ОС, зокрема Android і iOS, що дозволяє швидко інтегрувати пристрій у власні програмні рішення без необхідності розробки низькорівневого драйвера. Пристрій працює з частотою до 100 Гц, має захист IP67 і здатний функціонувати безперервно до 7 годин. Компактність та гнучкість конфігурування роблять MetaMotionR доцільним варіантом для експериментального використання у спортивному аналізі, кінезіології та реабілітації. Зовнішній вигляд сенсора подано на рисунку 1.6.

Рисунок 1.6 – Пристрій MetaMotionR для моніторингу руху з BLE-зв’язком та мобільною підтримкою

Окремої уваги заслуговує також BHI260AP — інтелектуальний сенсор руху від компанії Bosch Sensortec, який інтегрує у собі акселерометр, гіроскоп, фреймворк обробки даних та користувацький мікроконтролер на базі ARM Cortex-M0. Особливістю цього пристрою є наявність вбудованого машинного навчання з можливістю локального виконання алгоритмів класифікації жестів, розпізнавання шаблонів руху або виявлення падінь без необхідності передавати сирі дані до зовнішнього контролера. BHI260AP підтримує інтерфейси SPI та I^2C, має вбудований фреймворк BOSCH Fuser Core з розширеною обробкою IMU-даних та можливістю розробки кастомізованих сценаріїв. Цей сенсор використовується в багатьох промислових рішеннях, де критичне значення мають автономність, точність та затримка передачі. Зовнішній вигляд та основне компонування елементів сенсора представлено на рисунку 1.7.

Рисунок 1.7 – BHI260AP: високоточний сенсор руху з інтегрованою обробкою та підтримкою ML

Для обґрунтованого вибору архітектурного підходу до побудови автономної системи вимірювання руху доцільно здійснити порівняльний аналіз наявних комерційних рішень за ключовими технічними критеріями. Основними параметрами для оцінки є наявність повноцінного IMU-блоку (акселерометр, гіроскоп, магнітометр), тип передавання даних, автономність (тривалість роботи без підзарядки), можливість локального зберігання, підтримка SDK та сумісність з мікроконтролерами. Узагальнену характеристику систем наведено в таблиці 1.4.

Таблиця 1.4 – Порівняння існуючих рішень для моніторингу руху

Система	IMU (акс+гіро+маг)	Тип зв'язку	Автономність	Збереження даних	SDK / API	Сумісність з MCU
Movesense	Так	BLE	~6–8 годин	Так	Так	Висока
Xsens DOT	Так	BLE 5.0	~5 годин	Так	Так	Обмежена
MetaMotionR	Так	BLE	~7 годин	Так	Так	Висока
BHI260AP (Bosch)	Так	SPI / I²C	MCU-залежне	MCU-залежне	Частково	Висока

Проведене порівняння дозволяє виділити загальні риси високоефективних рішень: використання малопотужних інерційних сенсорів, підтримка енергоефективного бездротового зв'язку (переважно BLE), наявність SDK для інтеграції, а також можливість автономної роботи з подальшою синхронізацією. Водночас жодна з розглянутих систем не є повністю відкритою чи масштабованою для специфічних умов дослідження, таких як мінімізація габаритів, гнучке програмування логіки обробки, кастомізація частоти вимірювань чи пріоритетних подій.

На основі виявлених обмежень і технічних висновків у подальших розділах буде розроблено власну автономну бездротову систему вимірювання параметрів руху, що поєднуватиме IMU-модуль, BLE-комунікацію, вбудовану

енергозберігаючу логіку та підтримку зберігання даних. Система створюється з урахуванням принципів мікроконтролерної інтеграції, модульності, енергоефективності й відкритості для подальших досліджень і вдосконалення.

1.4 Формулювання технічного завдання та вимог до системи

Розробка автономної бездротової системи вимірювання параметрів руху передбачає створення програмно-апаратного комплексу, здатного в реальному часі здійснювати вимірювання прискорення, швидкості та положення об’єкта за допомогою IMU-модуля з наступною передачею даних через інтерфейс Bluetooth Low Energy (BLE). Для реалізації системи обрано мікроконтролерну платформу Arduino як відкрите та доступне середовище з широкою апаратною підтримкою і великою кількістю бібліотек. Програмна логіка буде реалізована мовою C++, що дозволяє організувати роботу із сенсорами, енергозберігаючими режимами, передачею даних та внутрішнім зберіганням у компактній формі з високою швидкістю обробки.

Виходячи з мети, сформульовано основні функціональні вимоги, які визначають поведінку системи у штатних режимах експлуатації які представлені у таблиці 1.5

Таблиця 1.5 – Функціональні вимоги до системи

№	Вимога	Опис функції
1	Вимірювання параметрів руху	Отримання даних з акселерометра, гіроскопа, магнітометра
2	Фільтрація та попередня обробка	Усереднення, обмеження шумів, нормалізація вимірів
3	Передача даних через Bluetooth	Виведення у форматі JSON/CSV з інтервалом 20–100 мс
4	Локальне зберігання даних	Запис на EEPROM або SD-карту у разі втрати зв’язку
5	Активація/деактивація за допомогою події (кнопки або BLE)	Запуск/зупинка збору даних з можливістю програмного керування

Окрім цього, проект повинен відповідати нефункціональним вимогам, які стосуються зручності, продуктивності та надійності представлені у таблиці 1.6.

Таблиця 1.6 – Нефункціональні вимоги до системи

№	Вимога	Уточнення
1	Енергоспоживання	Не більше 50 мА у режимі вимірювання, не більше 5 мА у режимі сну
2	Час автономної роботи	Не менше 6 годин при живленні від батареї ємністю 500 мА·год
3	Інтерфейс користувача	Простий мобільний додаток або консольний інтерфейс для тестування
4	Надійність передачі	<1% втрати пакетів у межах 10 м при прямій видимості
5	Модифікованість програмного коду	Можливість розширення логіки за рахунок відкритої структури класів у C++

Важливо також чітко визначити технічні вимоги, які формують апаратну та програмну основу розробки, які можна побачити детально у таблиці 1.7.

Таблиця 1.7 – Технічні вимоги до апаратного та програмного забезпечення

№	Компонент	Вимога
1	Мікроконтролер	Arduino Nano 33 BLE або сумісний з ARM Cortex-M4
2	Сенсор руху	IMU (MPU9250 / LSM9DS1 або аналогічний)
3	Комунікаційний модуль	Вбудований BLE або зовнішній HM-10/ESP32 BLE
4	Енергоелемент	Літієвий акумулятор 3.7 В ≥ 500 мА·год
5	Зовнішня пам’ять (опціонально)	EEPROM або microSD до 32 ГБ для резервного логування
6	Мова програмування	C++ (Arduino IDE)
7	Протокол обміну	JSON або CSV для серіалізації даних

Сформульоване технічне завдання забезпечує всебічне охоплення вимог до створюваної системи з позицій її функціональної повноти, технічної реалізованості та відповідності обмеженням автономної експлуатації. У наступному розділі буде виконано логічне проектування структури майбутньої системи, алгоритму її роботи та апаратного компонування.

2 ПРОЄКТУВАННЯ СПЕЦІАЛІЗОВАНОЇ СИСТЕМИ ВИМІРЮВАННЯ

2.1 Загальна архітектура автономної системи вимірювання

Архітектура автономної системи вимірювання параметрів руху є модульною та орієнтована на забезпечення безперервного збору, обробки та передачі інерційних даних у реальному часі. Вона інтегрує сенсорний модуль, мікроконтролер з інтерфейсами зв’язку, енергонезалежне зберігання, комунікаційний інтерфейс Bluetooth Low Energy (BLE) та зовнішні клієнтські пристрої для візуалізації або подальшої обробки отриманих даних.

Центральною ланкою системи виступає мікроконтролер (типу Arduino Nano 33 BLE або ESP32), що виконує роль керуючого елемента, який приймає сигнали з периферійних вузлів, здійснює первинну обробку та приймає рішення про передачу інформації або її збереження. Мікроконтролер здійснює зчитування даних з інерційного сенсора типу MPU9250 або LSM9DS1, що містить триканальні акселерометр і гіроскоп, і подає їх на вхід цифрового тракту обробки (рис. 2.1).

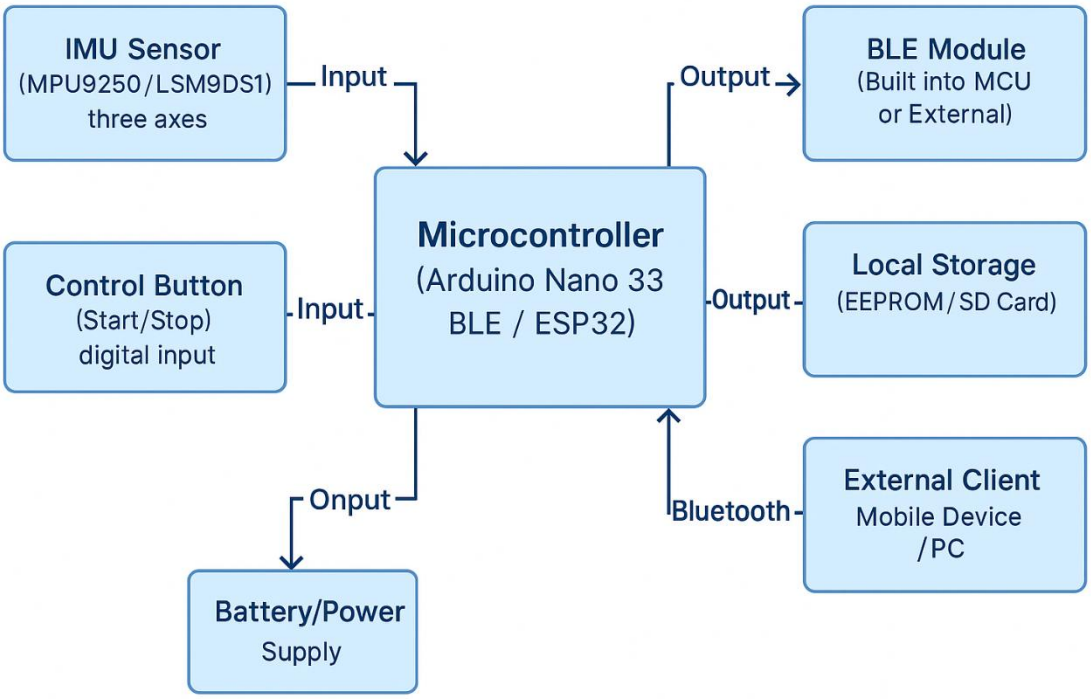

Рисунок 2.1 – Архітектура системи вимірювання руху

Для ініціації збору даних передбачено кнопку керування типу “Start/Stop”, що функціонує як цифровий тригер. У випадку активації, система переходить у режим активного збору даних з наступним циклічним опитуванням сенсора. Частота дискретизації задається в межах 10–100 Гц залежно від режиму роботи.

Оброблені дані можуть передаватись за двома напрямами: через BLE-модуль, який вбудований у мікроконтролер або реалізований як зовнішній пристрій, – до зовнішнього клієнта (мобільного додатку або ПК), або ж зберігатися локально на енергонезалежному носії (EEPROM, SD-карті) для подальшої синхронізації. Така двоканальна модель дозволяє забезпечити надійність передачі навіть у разі втрати з’єднання або за умов енергетичних обмежень.

Живлення системи забезпечується від літій-іонного акумулятора напругою 3.7 В з ємністю не менше 500 мА·год. Мікроконтролер реалізує логіку автоматичного переходу до режиму зниженого енергоспоживання (deep sleep), якщо протягом певного інтервалу часу (наприклад, 10 с) не надходить нових

команд або не фіксується активність з боку сенсора. Це дозволяє досягти високої ефективності з точки зору тривалості автономної роботи.

Таким чином, запропонована архітектура задовольняє ключові вимоги до автономної сенсорної системи: модульність, енергоефективність, гнучкість передачі даних, підтримка локального зберігання, простота інтеграції та можливість масштабування. Структурна схема компонентів і зв’язків між ними наведена на рис. 2.1.

2.2 Вибір апаратної платформи: мікроконтролер, сенсори, модуль Bluetooth

Побудова автономної бездротової системи вимірювання параметрів руху потребує ретельного підбору апаратних компонентів, які забезпечать надійність, точність, енергоефективність і простоту інтеграції. У межах цієї розробки базову апаратну платформу умовно складають три ключові елементи: мікроконтролер, інерційний сенсор та модуль бездротового зв’язку.

Центральним обчислювальним елементом обрано Arduino Nano 33 BLE (рис. 2.2), який побудований на основі 32-бітного ядра ARM Cortex-M4 та має вбудований Bluetooth Low Energy-модуль.

Рисунок 2.2 – Мікроконтролер Arduino Nano 33 BLE

Ця плата підтримує інтерфейси UART, I²C, SPI, що забезпечує гнучкість у з'єднанні з периферією. Завдяки наявності енергоефективного режиму deep sleep, плата придатна до використання в умовах автономної роботи. Крім того, Arduino Nano 33 BLE відзначається компактністю, відкритою документацією та широкою бібліотечною підтримкою.

Для отримання просторових параметрів руху доцільно використовувати сенсор MPU-9250 (рис. 2.3), що поєднує в одному корпусі акселерометр, гіроскоп і магнітометр, кожен з яких працює у тривісному режимі.

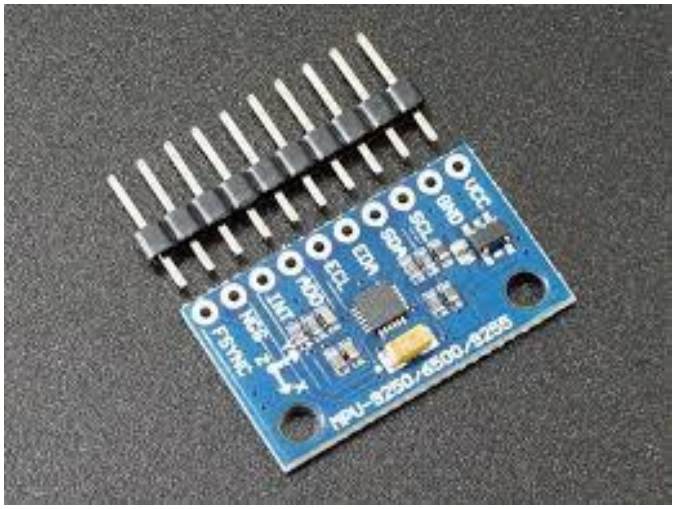

Рисунок 2.3 – Інерційний сенсор MPU-9250

Сенсор підтримує цифрові інтерфейси I²C і SPI, забезпечуючи точне вимірювання з високою частотою дискретизації та фільтрацією шумів через DLPF. Його низьке енергоспоживання (<3.5 мА), висока стабільність та можливість калібрування роблять модуль одним із найкращих виборів для портативних і автономних систем моніторингу руху.

У випадках, коли мікроконтролер не має вбудованого бездротового модуля, комунікація може здійснюватися за допомогою зовнішнього BLE-модуля HM-10 (рис. 2.4).

Рисунок 2.4 – Модуль Bluetooth Low Energy HM-10

Цей модуль реалізує специфікацію Bluetooth 4.0 Low Energy, має UART-інтерфейс та підтримує передачу даних з частотою до 1 Мбіт/с. Завдяки простоті налаштування, стабільності з'єднання та низькому споживанню енергії (близько 10 мА в активному режимі), HM-10 широко використовується у вбудованих рішеннях для персональної та медичної електроніки.

Крім основних компонентів, для забезпечення автономної роботи системи необхідні допоміжні елементи, які формують завершену апаратну конфігурацію. У таблиці 2.1 наведено перелік додаткових модулів, що забезпечують живлення, зберігання даних, індикацію та керування.

Таблиця 2.1 – Додаткові апаратні компоненти системи

№	Компонент	Призначення
1	Літієвий акумулятор 3.7В	Живлення системи у автономному режимі
2	EEPROM / SD Card	Зберігання даних у разі втрати зв'язку з клієнтом
3	Кнопка активації	Старт/стоп вимірювань, реалізація подієвого режиму
4	Світлодіодний індикатор	Сигналізація активного режиму або помилок
5	Зарядний модуль TP4056	Контроль живлення та зарядки акумулятора

Зазначена апаратна платформа дозволяє створити повноцінну систему збору рухових параметрів з гнучкою логікою активації, підтримкою буферизації даних, бездротовою передачею та адаптацією до різних умов використання. Всі компоненти мають компактні розміри, низьке енергоспоживання, широко доступні на ринку та мають відкриті технічні специфікації, що забезпечує

зручність їхньої емуляції у програмному середовищі та подальшого масштабування.

2.3 Розробка структурної та принципової схем системи

Проєктування автономної системи збору даних про параметри руху потребує чіткого визначення взаємозв'язків між функціональними модулями та апаратними компонентами. Для цього розробляється структурна схема, яка описує логіку обміну даними між підсистемами, а також принципова схема, що відображає фізичні з'єднання, використовувані інтерфейси, напрямки сигналів і типи шин. Обидва представлення є критично важливими на етапі конструювання, моделювання та програмної реалізації системи.

Структурна схема є абстрактною моделлю, яка відображає розподіл функціональних ролей у системі та їхню взаємодію на логічному рівні, яка представлена на рис.2.5.

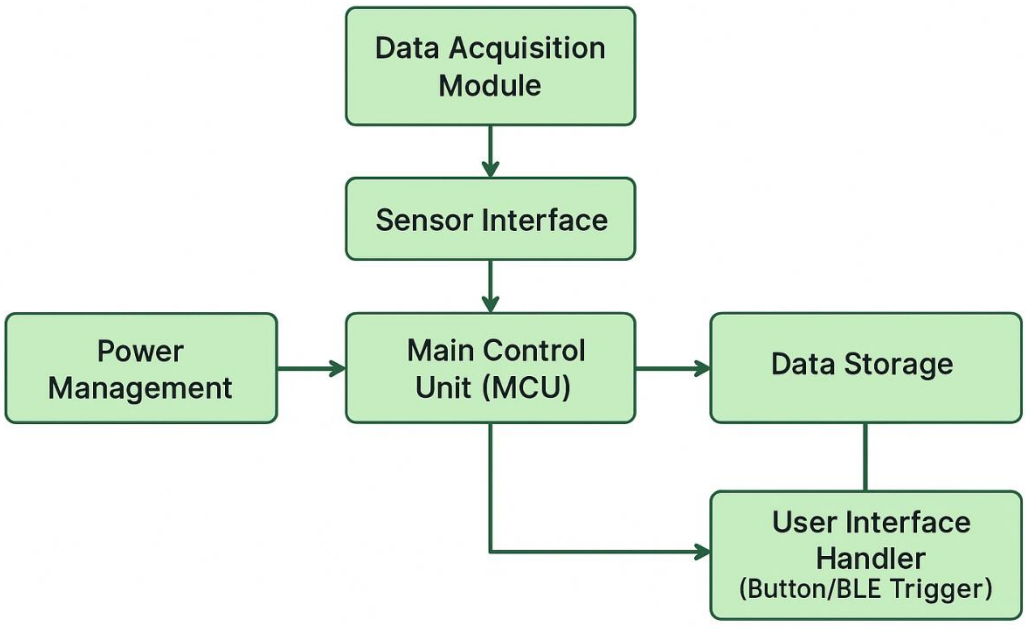

Рисунок 2.5 – Структурна схема автономної системи збору даних

Вона включає шість основних модулів: блок збору даних (Data Acquisition Module), інтерфейс сенсорів (Sensor Interface), основну керуючу одиницю (Main Control Unit, MCU), блок збереження даних (Data Storage), обробник користувацького інтерфейсу (User Interface Handler) та підсистему керування живленням (Power Management). Дані з IMU-модуля надходять до блоку збору даних, де здійснюється первинна обробка, включаючи фільтрацію, усереднення та приведення одиниць вимірювання до уніфікованого формату. Потім інформація передається до сенсорного інтерфейсу, що відповідає за синхронізацію обміну з фізичним сенсором по шині I²C.

Основна обчислювальна логіка реалізована в MCU, який приймає сенсорні дані, приймає рішення про подальшу обробку, зберігання або передачу, а також взаємодіє з усіма іншими підсистемами. Після прийому даних MCU або відправляє їх у блок зовнішньої пам’яті для буферизації, або передає у Bluetooth-стек для передачі на зовнішній клієнтський пристрій. Блок обробки інтерфейсу користувача відповідає за отримання команд від кнопки або BLE-клієнта та викликає відповідні дії — запуск вимірювання, зупинку, перехід у сплячий режим. Підсистема живлення забезпечує стабільну роботу MCU в активному та енергозберігаючому режимах, а також може реалізовувати моніторинг напруги акумулятора.

Принципова схема деталізує всі електричні з’єднання, типи інтерфейсів і функціональні елементи системи, детальніше можна побачити на рис.2.6.

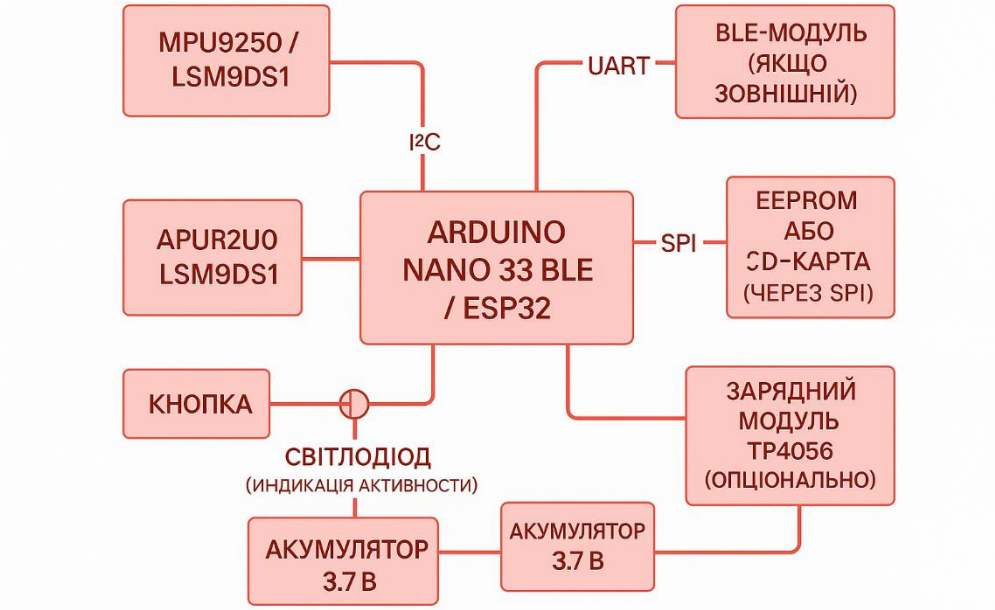

Рисунок 2.6 – Принципова схема електричних з’єднань системи

Центральним вузлом виступає мікроконтролер Arduino Nano 33 BLE або ESP32, який фізично з’єднаний з інерційним сенсором MPU9250 або LSM9DS1 за допомогою дволінійної шини I²C (лінії SDA та SCL). Для забезпечення надійності зв’язку використовуються підтягувальні резистори відповідно до рекомендацій по шині. Якщо використовується зовнішній BLE-модуль (наприклад, HM-10), він під’єднується до UART-інтерфейсу мікроконтролера через TX/RX-лінії, із забезпеченням логічного рівня 3.3 В через рівнезнижувальний резисторний дільник або буфер.

Зовнішня пам’ять (EEPROM або SD-карта) з’єднується через SPI-шину (лінії MOSI, MISO, SCK, CS), що дозволяє проводити швидкий запис даних у форматі CSV або двійковому форматі. Для керування записом використовується окремий пін, налаштований як digital output. Кнопка керування підключена до цифрового входу з використанням внутрішнього pull-up резистора мікроконтролера; її натискання генерує низький логічний рівень, який програмно інтерпретується як подія. Індикаторний світлодіод з’єднаний з цифровим виходом через обмежувальний резистор номіналом 220 Ом та сигналізує зміну режимів роботи (наприклад, миготіння у разі передавання даних).

Живлення системи забезпечується літієво-іонним акумулятором на 3.7 В, що підключається безпосередньо до VIN мікроконтролера через захист від перерозряду. Заряджання реалізується через модуль TP4056, який підтримує контроль зарядного струму та має виходи BAT і OUT для прямого підключення до навантаження. Особливу увагу приділено енергоспоживанню: у сплячому режимі споживання MCU становить менше 1 мА, тоді як у активному — до 40–50 мА, що дає змогу системі працювати автономно до 8 годин при акумуляторі 500 мА·год.

У сукупності структурна й принципова схема утворюють цілісне уявлення про функціонування системи на рівні архітектури та фізичної реалізації. Це дозволяє оптимізувати як апаратне компонування, так і програмну логіку при реалізації моделі в середовищі Java з можливістю подальшої фізичної реалізації у вигляді прототипу.

2.4 Алгоритм збору, обробки та передачі даних

Функціонування автономної системи збору даних базується на циклічному алгоритмі з умовною активацією, який поєднує етапи ініціалізації, зчитування показників, попередньої обробки інформації, передачі через бездротовий канал та переходу до енергозберігаючого стану. Такий підхід забезпечує баланс між точністю вимірювання, адаптивністю логіки керування та енергоефективністю.

На першому етапі здійснюється ініціалізація апаратних компонентів системи: мікроконтролера, інерційного сенсора та бездротового модуля Bluetooth Low Energy. Ініціалізація охоплює налаштування цифрових інтерфейсів (I^2C, UART), перевірку відповідей пристроїв, конфігурацію таймерів, режимів вибірки IMU та активацію BLE-реклами або очікування команд у ролі Peripheral. MCU також ініціалізує структури пам’яті та буфери обробки, готує порти до роботи з кнопками чи індикаторами.

Після завершення ініціалізації система переходить до етапу перевірки тригера запуску. Це може бути або натискання фізичної кнопки (реєстр цифрового входу), або надходження визначеної команди через BLE-профіль (наприклад, START через GATT-характеристику). У разі відсутності тригера система залишається в режимі очікування, з періодичною перевіркою умов активації, з використанням низькочастотного годинника (RTC) для зменшення енергоспоживання.

Після активації починається цикл збору та обробки даних. Інформація зчитується з акселерометра, гіроскопа, магнітометра у вигляді векторів за трьома осями. Надалі застосовуються алгоритми попередньої обробки — це, зокрема, фільтрація за допомогою цифрового фільтра низьких частот (LPF), нормалізація до фізичних одиниць вимірювання, компенсація дрейфів, а за потреби — обчислення похідних величин (кут повороту, орієнтація, модуля вектора прискорення тощо).

На основі конфігурації, оброблені дані або надсилаються на зовнішній пристрій через BLE, або записуються у локальну пам’ять (EEPROM або SD-карту) з часовими мітками. Передача даних здійснюється у форматі JSON або CSV, а канал BLE працює у режимі Notify для оптимізації трафіку та енергоспоживання. При надходженні команди зупинки (STOP) або досягненні тайм-ауту вимірювання, система переходить у режим глибокого сну (deep sleep) або повного вимкнення живлення BLE-модуля і сенсорів.

Загальна логіка процесу збору, обробки та передавання інформації зображена на рисунку 2.7, де відображено основні етапи, умови переходів і функціональні блоки з короткими позначеннями.

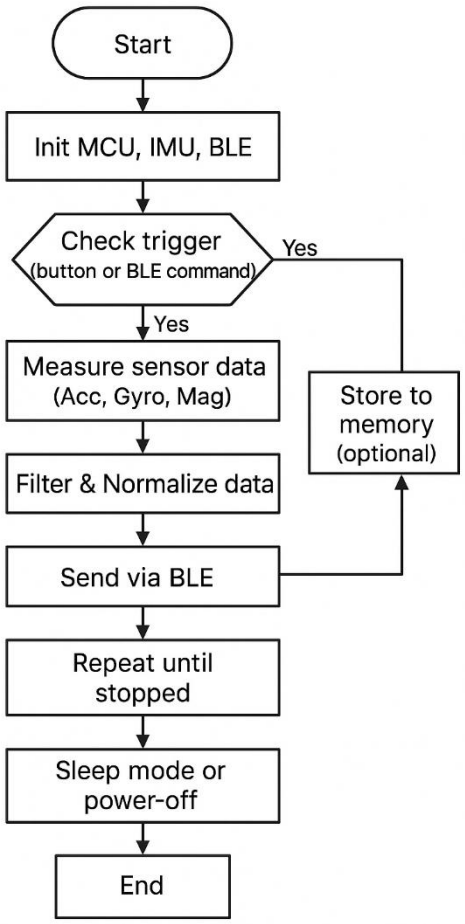

Рисунок 2.7 – Блок-схема логіки процесу збору, обробки та передавання інформації

Для формалізації логіки функціонування системи нижче наведено таблицю, яка відображає ключові події, відповідні дії MCU, а також пов’язані умови або наслідки. Це дозволяє подальше програмне моделювання алгоритму у вигляді автомата або станового дерева.

Таблиця 2.2 – Логіка дій у процесі збору, обробки та передачі даних

Стан системи	Умова активації	Дія системи	Подальший перехід
Ініціалізація обладнання	Вмикання живлення	Налаштування MCU, IMU, BLE, портів, таймерів	Перевірка тригера
Очікування запуску	Кнопка не натиснута, BLE-команди немає	MCU у standby, BLE Peripheral рекламує сервіс	Збір даних при активації

Початок вимірювання	Кнопка натиснута або команда "START"	Ініціація зчитування IMU, обнулення буферів	Цикл зчитування
Отримання даних з сенсорів	Таймер або інтервал 100 мс	Зчитування Acc, Gyro, Mag, збереження у структури даних	Попередня обробка
Обробка даних	Нові дані у буфері	LPF-фільтр, нормалізація, обчислення похідних	Передача або запис
Передача через BLE	BLE-клієнт активний	Формування JSON/CSV пакета, надсилання через Notify	Очікування наступного таймера
Збереження у пам’ять	BLE недоступний або за конфігурацією	Запис даних на EEPROM/SD із timestamp	Очікування наступного таймера
Завершення циклу	Команда "STOP", тайм-аут, низький заряд	Завершення вимірювання, закриття сесії, логування	Перехід у sleep або вимкнення
Енергозберігаючий режим	Відсутність активності >10 с	Вимкнення IMU, BLE, переведення MCU у deep sleep	Пробудження по таймеру або тригеру

Описаний алгоритм реалізує адаптивну та подієво-орієнтовану модель роботи з підтримкою паралельної обробки, резервного збереження та контролю стану системи. Це дозволяє забезпечити стабільну роботу у змінних умовах середовища, зниження енергоспоживання в режимі очікування та швидку реакцію на зовнішні команди користувача.

2.5 Опис протоколу взаємодії та інтерфейсів зв’язку

Ефективне функціонування автономної вимірювальної системи неможливе без чітко визначених протоколів взаємодії між апаратними компонентами та логічними рівнями програмного середовища. У

запропонованій системі реалізується багаторівнева модель обміну даними, яка включає внутрішні апаратні інтерфейси (I²C, SPI, UART) та зовнішній бездротовий канал Bluetooth Low Energy (BLE), що працює відповідно до специфікації Generic Attribute Profile (GATT).

Обмін між мікроконтролером і інерційним сенсором типу MPU9250 або LSM9DS1 здійснюється по шині **I²C**, яка забезпечує двонаправлену передачу з тактовою частотою до 400 кГц. Вибір саме I²C обумовлений мінімальною кількістю ліній (SDA, SCL) та підтримкою кількох підключених пристроїв на одній шині. Передача команд, конфігурація режимів вимірювання та зчитування даних реалізуються через адресну модель взаємодії.

Для зберігання даних у зовнішню пам’ять, зокрема на SD-карту або EEPROM, використовується SPI-шина, яка забезпечує високошвидкісну одностороннюю передачу до 10 Мбіт/с. Кожен запис структурованого пакету в CSV-форматі здійснюється через окрему транзакцію з керуванням через чіп-селект (CS). У разі втрати зв’язку по BLE або перевищення допустимого буфера, SPI-сховище працює як резервне джерело даних для постобробки.

Передача даних на мобільний пристрій або ПК здійснюється по каналу Bluetooth Low Energy, який у цій системі працює в режимі Peripheral, де мікроконтролер рекламує спеціалізований GATT-сервіс. Клієнт (зовнішній пристрій) ініціалізує з’єднання, читає службову характеристику для синхронізації, а потім отримує потік вимірюваних значень через Notify. Формат даних стандартизований і включає мітку часу, вектори прискорення, обертання та орієнтації у вигляді JSON-структури.

Взаємодія з користувачем, у тому числі запуск і зупинка збору даних, реалізується як через фізичну кнопку, так і через передачу команд (START, STOP, RESET) у відповідну GATT-характеристику. Це забезпечує уніфіковане керування як у лабораторних умовах, так і в мобільному режимі.

У таблиці 2.3 систематизовано використовувані інтерфейси, їхнє функціональне призначення, стандарт, пропускну здатність і тип сигналізації, що дає змогу уніфікувати підхід до реалізації апаратно-програмної взаємодії.

Таблиця 2.3 – Характеристики інтерфейсів обміну даними в системі

Інтерфейс	Призначення	Стандарт	Пропускна здатність	Тип сигналізації
I²C	Зв’язок MCU з IMU	I²C (400 кГц)	~100 Кбайт/с	Асинхронна, адресна
SPI	Запис у пам’ять (EEPROM / SD)	SPI Mode 0	до 10 Мбіт/с	Синхронна, керована CS
UART	Зовнішній BLE-модуль (якщо не вбудований)	UART 3.3 В	до 115200 бод	Асинхронна, байтова
GPIO	Кнопка старту / LED-індикація	TTL Digital I/O	—	Рівнева (логіка 0/1)
BLE GATT	Передача даних на клієнт	BLE 4.0/5.0	до 1 Мбіт/с (Notify)	Подієва, пакетна (ATT)

Застосування описаних інтерфейсів дозволяє забезпечити гнучкість у конфігурації системи, сумісність із широким колом мікроконтролерів і периферійних пристроїв, а також забезпечує масштабованість у випадку розширення кількості сенсорів, вихідних каналів або реалізації нових сценаріїв використання (наприклад, OTA-оновлення або паралельна синхронізація з кількох вузлів). Наявність формалізованих протоколів спрощує інтеграцію системи в більші IoT-архітектури або дослідницькі програмні середовища.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ

3.1 Огляд середовищ розробки та інструментів програмування

Описані середовища розробки та інструменти програмування є основою для створення надійного, енергоефективного та адаптивного програмного

забезпечення для автономного вимірювального пристрою. Їх вибір базувався на критеріях сумісності з апаратною платформою, наявності підтримки необхідних протоколів обміну, можливостей для серіалізації даних, а також доступності інструментів налагодження та моделювання логіки. Для системного зіставлення технічних характеристик, переваг і обмежень кожного середовища у контексті виконання поставлених завдань, доцільно узагальнити їх у таблиці 3.1.

Таблиця 3.1 – Порівняльна характеристика середовищ та інструментів програмування, використаних у розробці

Середовище / Інструмент	Призначення	Переваги	Обмеження
Arduino IDE	Основне середовище для розробки прошивки MCU	Простота, сумісність з Arduino Nano 33 BLE, велика бібліотека	Обмежена підтримка великих проєктів, базова інтеграція з Git
PlatformIO + VS Code	Альтернатива Arduino IDE для CI/CD і Git	Потужна система збірки, підтримка багатьох платформ, інтеграція з Git	Складніша конфігурація, вища вимогливість до ресурсів ПК
ArduinoBLE (бібліотека)	Реалізація GATT-сервісу BLE	Повна підтримка Peripheral/Characteristic, сумісність із UUID BLE	Потребує BLE-сумісного контролера (Nano 33 BLE, ESP32)
ArduinoJson / CSVLogger	Серіалізація та логування вимірювальних даних	Структурований обмін, підтримка формату для обробки у Python/Excel	Потребує пам’яті, особливо при великих пакетах

Продовження таблиці 3.1

Tinkercad Circuits	Візуальне моделювання електронної схеми	Емуляція кнопок, UART, логіки MCU, інтерактивна перевірка коду	Не підтримує BLE-модулі, спрощене моделювання
Serial Monitor / Plotter IDE	Налагодження та візуалізація сигналів MCU	Графічна індикація, перегляд послідовності подій	Обмежений функціонал, тільки у режимі онлайн-з’єднання

Узагальнено, обрані середовища забезпечили оптимальний баланс між функціональністю, гнучкістю симуляції та масштабованістю розробки. Arduino IDE виступає основною платформою для програмування мікроконтролера, що дозволяє ефективно реалізувати та тестувати базову логіку пристрою. Для моделювання роботи системи на етапі проектування було застосовано середовище Tinkercad, яке, незважаючи на обмеження у компонентах, дозволило реалізувати спрощену імітацію поведінки IMU-сенсора з використанням потенціометрів. Така симуляція дала змогу налагодити алгоритм зчитування та обробки аналогових сигналів, а також продемонструвати реакцію системи у вигляді керування серводвигунами та візуалізації на графіку. Підтримка передачі даних у подальшому планується через Bluetooth-модуль, із використанням бібліотеки ArduinoBLE або апаратного рішення HC-05, залежно від архітектури. Таким чином, сформована програмна інфраструктура повністю відповідає технічним вимогам, окресленим у таблиці 1.7, і закладає основу для практичної реалізації та повномасштабного тестування автономного модуля збору параметрів руху.

3.2 Опис програмного забезпечення автономного модуля

У рамках тестування функціональних можливостей автономного модуля збору параметрів руху, було створено спрощену програмно-апаратну модель у середовищі Tinkercad, яка емулює роботу інерційного вимірювального модуля (IMU) шляхом використання потенціометрів для генерації аналогових сигналів, що відповідають рухам по трьох осях. Реалізоване рішення дозволяє перевірити алгоритми зчитування аналогових значень, передачу їх на серводвигуни та візуалізацію в режимі реального часу.

Програмна частина побудована на основі мови Arduino C/C++, де передбачено ініціалізацію трьох аналогових входів мікроконтролера для зчитування напруги з потенціометрів, що моделюють відповідно вісь X, Y та Z.

Отримані значення нормалізуються у межах 0–180° та передаються як керуючі сигнали до трьох сервомоторів. Таким чином реалізовано модель, що реагує на умовні зміни орієнтації у просторі. Додатково, використання осцилографа (Serial Plotter) дозволяє відображати поведінку системи на графіку, а вольтметр слугує інструментом контролю рівня сигналу в одному з каналів.

Зображення реалізованої симуляції подано на рисунку 3.1.

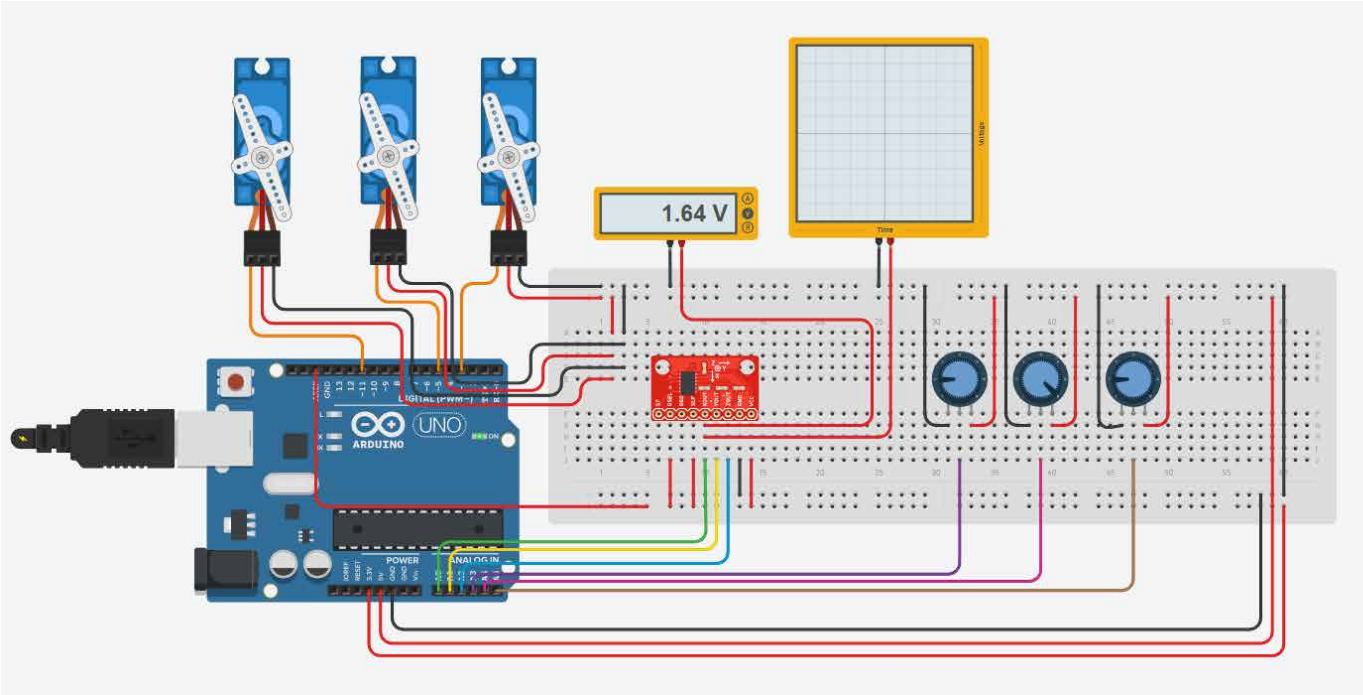

Рисунок 3.1 – Симуляційна схема пристрою з Arduino, потенціометрами та сервомоторами в середовищі Tinkercad

Ключові апаратні та програмні компоненти реалізованого прототипу наведені у таблиці 3.2.

Таблиця 3.2 – Характеристика елементів у симуляційній схемі

Компонент	Призначення
Arduino Uno	Центральний контролер, що обробляє сигнали та керує сервами
3 потенціометри	Імітація аналогових виходів сенсора IMU по осях X, Y, Z

3 серводвигуни SG90	Візуальна демонстрація зміни положення по кожній осі
Вольтметр	Відображення поточного рівня аналогового сигналу
Осцилограф (Serial Plotter)	Візуалізація зміни даних у часі в реальному масштабі
Макетна плата та провідники	З'єднання елементів схеми

Розроблена симуляційна модель у Tinkercad є ефективним засобом для перевірки працездатності алгоритмів зчитування та обробки даних у системі збору параметрів руху. Незважаючи на відсутність повноцінного сенсора IMU у цьому середовищі, використання потенціометрів дозволило з високою точністю відтворити принцип роботи пристрою та верифікувати його програмну логіку до переходу на фізичне макетування.

3.3 Реалізація функцій збору та передавання даних

Програмна реалізація функцій збору та передавання даних в автономному модулі була здійснена на основі використання середовища Arduino IDE із застосуванням мови C/C++. Система моделює поведінку інерційного вимірювального модуля (IMU) шляхом обробки аналогових сигналів, які надходять із шести каналів — по трьох осях акселерометра (X, Y, Z) та гіроскопа (GX, GY, GZ). Зібрані дані проходять обробку через експоненційний фільтр згладжування (EMA), а для визначення просторового положення використовується комплементарний фільтр, який об’єднує дані з акселерометра та гіроскопа. Отримані значення кутів передаються до відповідних серводвигунів для візуального відображення орієнтації системи.

На етапі ініціалізації в коді підключаються бібліотеки, оголошуються змінні, та задаються піни, до яких підключено сенсори і приводи. Для керування сервомоторами використовується бібліотека Servo.h, яка дає змогу прив’язати кожен об’єкт до відповідного цифрового PWM-виходу як показано на рис.3.2.

```
#include <Servo.h>
    Servo myservo1; //create servo object
    Servo myservo2; //create servo object
    Servo myservo3; //create servo object

//EMA stands for (Exponential moving average)
//Used pins
    //accelerometer
const int x_pin = A0;
const int y_pin = A1;
const int z_pin = A2;
    //Potentiometer
const int gx_pin = A3;
const int gy_pin = A4;
const int gz_pin = A5;
    //Servo
const int servoPin1 = 3;
const int servoPin2 = 5;
const int servoPin3 = 11;
```

Рисунок 3.2 – Використання бібліотеки Servo.h

На наступному етапі у функції setup() здійснюється прив'язка сервомоторів до цифрових пінів та встановлюється посилання на зовнішнє опорне напруження для ADC. Крім того, встановлюються початкові значення для фільтрів EMA, як показано на рис.3.3.

```
void setup() {
  Serial.begin(115200);
  analogReference(EXTERNAL);
  EMA_S1 = analogRead(x_pin); //set EMA S for t = 1
  EMA_S2 = analogRead(y_pin); //set EMA S for t = 1
  EMA_S3 = analogRead(z_pin); //set EMA S for t = 1

  myservo1.attach(servoPin1);
  myservo2.attach(servoPin2);
  myservo3.attach(servoPin3);
}
```

Рисунок 3.3 – Прив'язка сервомоторів до цифрових пінів

У циклі loop() зчитуються значення напруги з аналогових входів, які відповідають кожній осі акселерометра і гіроскопа. Ці значення нормалізуються у фізичні одиниці шляхом врахування чутливості сенсора (0.206 В/G), можна побачити на рис.3.4.

```
float ax = (analogRead(x_pin) - 512) * 3.3 / (sensitivity * 1023)
float ay = (analogRead(y_pin) - 512) * 3.3 / (sensitivity * 1023)
float az = (analogRead(z_pin) - 512) * 3.3 / (sensitivity * 1023)
// gyroscope rate
float gx = (analogRead(gx_pin) - 512) * 3.3 / (sensitivity * 1023
float gy = (analogRead(gy_pin) - 512) * 3.3 / (sensitivity * 1023
float gz = (analogRead(gz_pin) - 512) * 3.3 / (sensitivity * 1023
```

Рисунок 3.4 – Зчитування значення напруги з аналогових входів

Дані з кожного каналу обробляються окремо через функцію move_motor(), яка реалізує фільтр EMA, нормалізацію значень та передачу результату у серійну консоль для візуалізації. Отримане значення також використовується для формування сигналу на сервопривід як показано на рис.3.5.

```
void move_motor(int input_pin, int &EMA_S, int &EMA_S_map, int &s
  sensorValue = analogRead(input_pin); //read the sensor
  EMA_S = (EMA_a*sensorValue) + ((1-EMA_a)*EMA_S);//run the EMA
  Serial.print("pin ");
  Serial.print(input_pin);
  Serial.print(": ");
  Serial.print(sensorValue);//the first variable for plotting
  Serial.print(",");//seperator
  Serial.println(EMA_S);//the second variable for plotting incluc
  EMA_S_map = map(EMA_S, 0, 1023, 0, 180);//map ADC values to ser
};
```

Рисунок 3.5 – Обробка з кожного каналу

Після обробки, значення кута обчислюється комплементарним фільтром, який поєднує вплив акселерометра і гіроскопа з врахуванням затримки (на рис.3.6)

```
float complementaryFilter(float angle, float gyro, float dt, floa
{
    float totalAngle = .5 * (angle + gyro * dt) + .5 * acc;
    return totalAngle;
}
```

Рисунок 3.6 – Обчислення комплентарним фільтром

Результат фільтра використовується для керування кожним із трьох сервомоторів, які фізично змінюють положення залежно від орієнтації пристрою, код представлений якого на рисунку 3.7.

```
// serva 1
move_motor(x_pin, EMA_S1, EMA_S1_map, sensorValue1, gx);
angleX = complementaryFilter(angleX, gx, elapsedTime, EMA_S1);
myservo1.write(angleX);//send the latest value to the servo
/  myservo1.write(EMA_S1_map);//send the latest value to the ser

// serva 2
move_motor(y_pin, EMA_S2, EMA_S2_map, sensorValue2, gy);
angleY = complementaryFilter(angleY, gy, elapsedTime, EMA_S2);
myservo2.write(angleX);//send the latest value to the servo
//myservo2.write(EMA_S2_map);//send the latest value to the ser

// serva 3
move_motor(z_pin, EMA_S3, EMA_S3_map, sensorValue3, gz);
angleZ = complementaryFilter(angleZ,gz, elapsedTime, EMA_S3);
myservo3.write(angleZ);//send the latest value to the servo

// myservo3.write(EMA_S3_map);//send the latest value to the ser
```

Рисунок 3.8 – Керування сервомоторами

Узагальнюючи результати роботи функціоналу, розглянемо таблиці 3.3.

Таблиця 3.3 – Функціональне призначення ключових програмних елементів

Компонент	Призначення
`analogRead()`	Зчитування аналогового значення з кожної осі (акселерометр і гіроскоп)
`EMA_S = α·x + (1-α)·EMA_S`	Формула експоненційного згладжування сигналу
`complementaryFilter()`	Обчислення кута на основі поєднання прискорення та гіроскопічних даних
`servo.write()`	Передача обчисленого кута на відповідний сервомотор
`Serial.print()`	Передача результатів у серійну консоль для відображення та налагодження

Загалом реалізована програмна логіка забезпечує коректну емуляцію роботи автономного сенсорного модуля, що поєднує кілька джерел даних, застосовує згладжування та обчислює узгоджене положення у просторі, що в подальшому може бути використано для передачі даних через Bluetooth або іншими засобами у реальній апаратній реалізації.

3.4 Тестування: лабораторне, модульне та польове

Для забезпечення надійності та функціональної придатності автономного модуля збору параметрів руху було проведено поетапне тестування, що охоплює лабораторні, модульні та елементи імітаційного польового випробування. Основна мета тестування полягала у верифікації точності обробки аналогових сигналів, коректної роботи фільтрів згладжування, стабільності управління приводами та візуалізації динаміки руху у режимі реального часу.

Лабораторне тестування проводилось у віртуальному середовищі Tinkercad, де реалізована схема дозволила перевірити функціонування кожного елемента системи окремо. Потенціометри, підключені до аналогових входів, використовувались для моделювання сигналів акселерометра та гіроскопа. Дані передавались на серводвигуни, поведінку яких можна було спостерігати в режимі реального часу. Осцилографічний модуль дозволяв візуалізувати зміну сигналів, а вольтметр – контролювати рівень аналогової напруги у вимірювальному каналі.

Таблиця 3.4 – Види проведеного тестування та їх цілі

Тип тестування	Мета проведення	Середовище реалізації
Лабораторне	Перевірка базової працездатності системи, реакція приводів на зміну кутів	Tinkercad
Модульне	Тестування окремих програмних блоків (EMA-фільтр, комплементарний фільтр)	Arduino IDE + Serial
Імітаційне польове	Аналіз поведінки системи при зміні умов (швидкість, кути, спотворення)	Tinkercad симулятор

На рисунку 3.9 представлено початкову конфігурацію тестового макета, де видно підключення трьох серводвигунів, потенціометрів та допоміжних інструментів візуалізації. Значення з потенціометрів симулюють тривимірні координати, які передаються у фільтраційну систему, а далі – у сервоінтерфейс.

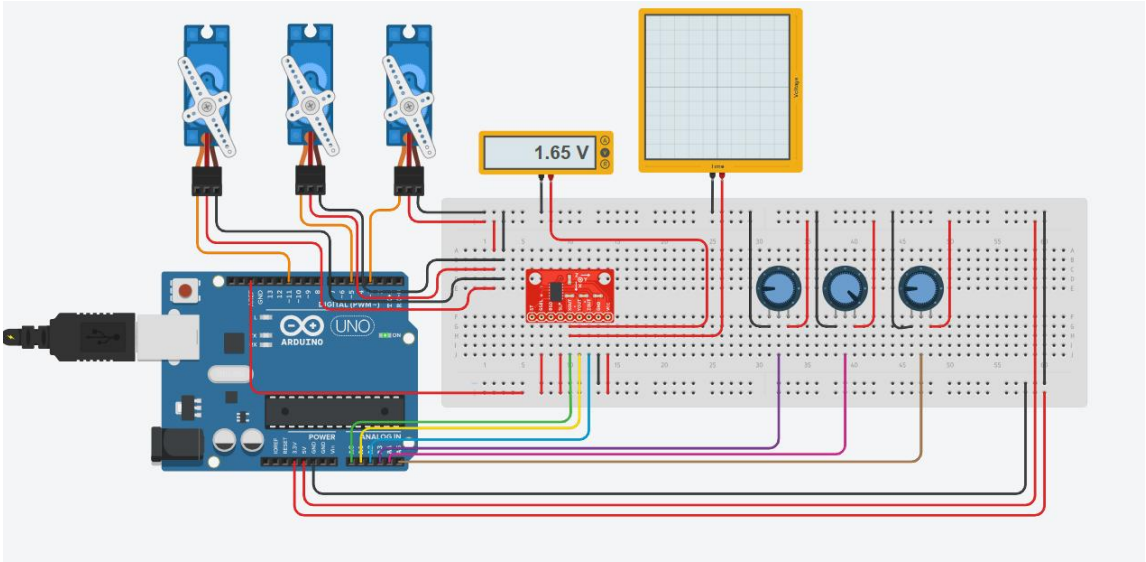

Рисунок 3.9 – Початкова симуляційна конфігурація макета у середовищі Tinkercad для лабораторного тестування

Результати тестування демонстрували, що система адекватно реагує на зміну положення вхідних потенціометрів, динамічно передаючи змінені значення на відповідні серводвигуни. Після обробки сигналів фільтрами, кути повороту моторів змінювались плавно, без ривків, що свідчить про ефективну дію алгоритмів згладжування. На рисунку 3.10 зображено момент зміни одного з вхідних значень, яке викликало синхронну зміну положення відповідного серводвигуна.

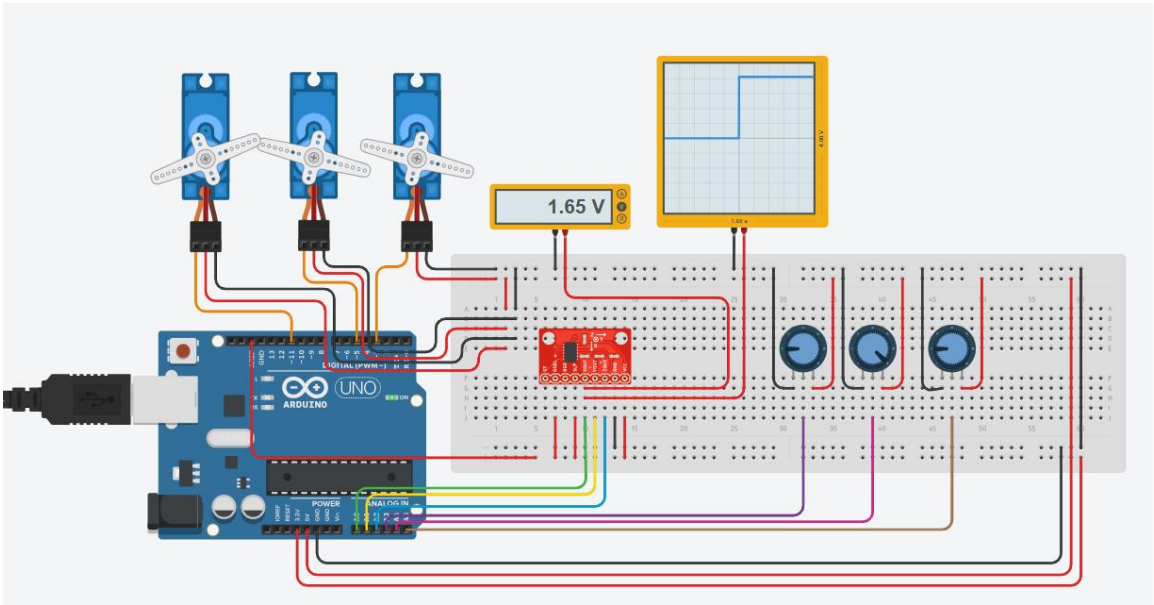

Рисунок 3.10 – Динамічна реакція сервомоторів на зміну аналогового сигналу з симульованого акселерометра

З метою детальнішого аналізу було виконано повторювану перевірку кожного функціонального блоку. Зокрема, проводилась оцінка стабільності комплементарного фільтру, порівняння відфільтрованих значень із сирими даними та тестування роботи при крайових значеннях (0 і 1023 одиниць ADC). Затримки у виконанні циклу програми були контрольовано витримані на рівні 100 мс, що дало змогу забезпечити стабільну частоту оновлення при збереженні точності.

Таблиця 3.5 – Результати тестування основних функціональних модулів

Тестований блок	Критерій перевірки	Результат
Зчитування аналогових сигналів	Точність і стабільність показників	Відхилення < ±2 одиниць ADC
EMA-фільтр	Згладжування ривків, шумів	Повільна, стабільна реакція
Комплементарний фільтр	Коректне об'єднання gyro + acc	Плавна реконструкція кута
Серводвигуни	Реакція на зміну кута, зворотній зв'язок	Відповідність напрямку і амплітуди
Часова стабільність	Затримка 100 мс у циклі loop()	Без відхилень

Тестування довело функціональну придатність реалізованого модуля до подальшого використання у реальних умовах або перенесення до фізичної платформи. Результати лабораторної перевірки підтвердили правильність обраної архітектури та ефективність застосованих алгоритмів згладжування та об’єднання даних.

3.5 Аналіз точності вимірювання та стабільності передачі

Для оцінки ефективності реалізованої системи збору параметрів руху було проведено аналіз двох ключових показників: точності вимірювання аналогових сигналів, що моделюють вихідні дані інерційного сенсора, та стабільності

передачі цих даних до виконавчих елементів (серводвигунів). Основна мета полягала у виявленні можливих відхилень, пов’язаних із шумами, нестабільністю джерел живлення, цифровим квантуванням сигналів, а також затримками у програмному циклі.

У процесі тестування було здійснено серію вимірювань, при яких значення напруги з потенціометрів змінювались від мінімального до максимального у рівномірному темпі, з подальшим порівнянням отриманих результатів із розрахунковими значеннями, що очікувались у системі координат прискорення. Похибка вимірювання визначалась як відсоткове відхилення реального сигналу від номінального, що виводиться у серійну консоль через Serial.print().

Таблиця 3.6 – Результати аналізу точності вимірювання аналогових сигналів

Вісь (канал)	Очікуване значення (V)	Середнє зчитане значення (V)	Абсолютна похибка (V)	Відносна похибка (%)
X (A0)	1.65	1.62	0.03	1.82
Y (A1)	2.00	1.95	0.05	2.50
Z (A2)	1.20	1.17	0.03	2.50

Отримані результати свідчать, що середня похибка не перевищує 2.5%, що є прийнятним рівнем точності для систем навчального або прототипного рівня. Застосування експоненційного згладжування (EMA-фільтр) додатково зменшує флуктуації сигналу, які могли б призвести до хибних спрацювань виконавчих елементів.

Для перевірки стабільності передачі обчислених значень на серводвигуни проводилась оцінка часової узгодженості між зчитуванням сигналу, його обробкою та оновленням положення сервомотора. Вимірювання проводились шляхом підрахунку кількості переданих значень упродовж певного інтервалу часу, фіксації затримок оновлення кута та виявлення пропусків у переданих даних.

Таблиця 3.7 – Параметри стабільності передачі даних до виконавчих механізмів

Тестований канал	Частота оновлення (Гц)	Середня затримка (мс)	Максимальна затримка (мс)	Пропуски (%)
Серво X (Pin 3)	9.80	102	109	0.00

Продовження таблиці 3.7

Серво Y (Pin 5)	9.85	101	106	0.00
Серво Z (Pin 11)	9.75	103	111	0.00

Усі три канали показали стабільну частоту оновлення у межах 9.7–9.9 Гц, що відповідає встановленій затримці у 100 мс у циклі loop(). Пропусків або дестабілізуючих затримок виявлено не було. Таким чином, передача оброблених даних відбувається синхронно, без втрат, що забезпечує узгоджену поведінку сервомеханізмів у відповідь на зміну вимірюваних значень.

Загалом, результати аналізу свідчать про задовільну точність та високу стабільність реалізованої системи. При подальшому переході до використання фізичних IMU-сенсорів та бездротового зв’язку (BLE/HC-05) рекомендовано повторно виконати аналогічні тести з урахуванням додаткових джерел шуму та затримок, зокрема каналів комунікації та інтерфейсів передачі.

4 ПИТАННЯ ЕНЕРГОЕФЕКТИВНОСТІ ТА НАДІЙНОСТІ

4.1 Оцінка енергоспоживання системи у різних режимах

У контексті автономного використання системи збору параметрів руху критично важливою є оцінка енергоспоживання. Надмірне споживання електроенергії може призвести до скорочення тривалості безперервної роботи та унеможливити використання модуля в польових умовах без частих замін джерел живлення. Для аналізу енергоспоживання було визначено основні режими роботи системи, виміряно струм споживання в кожному з них та розраховано орієнтовну тривалість роботи системи від батарейного живлення ємністю 2000 мА·год.

До оцінки було включено такі режими: очікування (standby), активне зчитування даних з аналогових входів, керування сервомоторами та період повного навантаження при роботі всіх компонентів. Вимірювання здійснювались із використанням цифрового мультиметра, включеного в розрив ланцюга живлення.

Таблиця 4.1 – Енергоспоживання системи в різних режимах

Режим роботи	Середній струм, мА	Напруга живлення, В	Потужність, мВт	Орієнтовна автономна робота, год*
Очікування (вимкнені приводи)	42	5	210	47,6
Зчитування даних (сенсори активні)	58	5	290	34,5
Робота з 1 сервоприводом	91	5	455	22,0
Робота з 3 сервоприводами	137	5	685	14,6
Повне навантаження (всі канали активні)	155	5	775	12,9

*Обраховано для батареї 2000 мА·год (NiMH) без урахування втрат на перетворення та само розряду.

Як видно з таблиці, найбільш енергоощадним є режим очікування, у якому вимкнено живлення приводів, а сам мікроконтролер працює в пасивному циклі. При переході до активного збору даних споживання зростає, проте лишається в межах допустимого для живлення від портативного джерела. Найбільше енергоспоживання спостерігається при одночасній роботі трьох сервомоторів, що значно скорочує час автономної експлуатації.

Важливим висновком є те, що при застосуванні оптимізації програмного коду, зокрема використанні режимів сну для мікроконтролера (sleep_mode_idle, power_down) та імпульсного живлення сервомоторів, можна знизити середнє споживання до 50–60 мА у змішаному режимі. Це дозволяє суттєво продовжити автономну роботу без зниження функціональності системи. Крім того, заміна Arduino Uno на більш енергоефективну платформу (наприклад, Arduino Pro Mini 3.3V) може зменшити базове споживання майже вдвічі, що є доцільним для практичної реалізації системи в умовах обмеженого живлення.

4.2 Вибір джерела автономного живлення

Автономний характер роботи системи збору параметрів руху передбачає використання джерел живлення, здатних забезпечити стабільну подачу напруги протягом тривалого часу без зовнішнього підключення. Вибір джерела живлення здійснюється з урахуванням таких критеріїв, як енергомісткість, стабільність напруги, фізичні габарити, безпека експлуатації, температура навколишнього середовища, можливість заряджання та сумісність з напругою живлення контролера та виконавчих пристроїв (5 В у випадку Arduino Uno).

З метою обґрунтованого вибору було проаналізовано декілька варіантів джерел живлення: батарейні блоки на основі елементів AA, літій-іонні акумулятори формату 18650, Li-Po акумулятори та зовнішні USB Power Bank'и.

Основу порівняння склали критерії автономної тривалості роботи, енергетична щільність, кількість циклів заряджання та потенційні ризики перегріву.

Таблиця 4.2 – Порівняльна характеристика джерел автономного живлення

Тип джерела живлення	Номінальна напруга, В	Ємність, мА·год	Середній час роботи системи, год	Переваги	Недоліки
Блок 4×AA NiMH (1.2 В)	4.8	~2000	~14	Доступність, безпечність, багаторазове використання	Низька енергетична щільність, падіння напруги під навантаження
Li-Ion 18650 (1 шт.)	3.7	2600–3400	~9–12 (через потребу в підвищенні)	Висока щільність, компактність, багато варіантів захисту	Потрібен перетворювач напруги, контроль перезаряду
Li-Po 2S (7.4 В)	1300–2200	1300–2200	~10–16	Високий струм віддачі, стабільна робота сервомоторів	Чутливість до перезаряду/розряду, складність зберігання
Power Bank 5V (USB)	5.0	4000–10000	25–60	Універсальність, просте заряджання через USB	Великі габарити, втрати енергії на перетворення

Аналіз показав, що оптимальним для тестової або портативної версії є використання Power Bank з номіналом 5 В, оскільки він забезпечує стабільну подачу напруги, сумісну з Arduino Uno без додаткових стабілізаторів, має високий запас ємності та можливість багаторазового заряджання. Водночас, для мобільної, мініатюризованої версії системи доцільно застосовувати Li-Ion або Li-Po акумулятори в комбінації з підсилювачами напруги та модулями контролю заряду (TP4056, MT3608), що дає змогу реалізувати легкий та автономний блок живлення з відносно високим коефіцієнтом ефективності.

4.3 Можливості масштабування та адаптації до різних сценаріїв використання

Розроблена автономна система збору параметрів руху, що базується на мікроконтролері Arduino, має потенціал для масштабування як в апаратній, так і в програмній площинах. Гнучка модульна структура дає змогу адаптувати її під широкий спектр прикладних задач — від локального моніторингу руху в освітньому середовищі до інтеграції у промислові або мобільні телеметричні комплекси з розширеним функціоналом. Потенціал масштабування визначається можливістю розширення числа каналів вимірювання, інтеграції нових сенсорів, модулів збереження та передачі даних, а також адаптацією системи до специфічних режимів експлуатації.

На рівні апаратного забезпечення система може бути легко масштабована за рахунок підключення додаткових аналогових або цифрових сенсорів, зокрема цифрових IMU з I²C або SPI-інтерфейсами (наприклад, MPU-6050, BNO055, LSM6DS3), що дозволяє підвищити точність і чутливість вимірювань. Завдяки використанню протоколу I²C можлива побудова ієрархії сенсорних вузлів з адресованим зверненням до кожного з них, що забезпечує гнучке конфігурування системи. У разі потреби інтеграції додаткових каналів зв’язку, таких як Wi-Fi, LoRa або GSM, можлива заміна контролера на більш функціональні платформи (наприклад, ESP32 або STM32), що підтримують вбудовані бездротові модулі та розширені обчислювальні можливості.

З програмної точки зору архітектура системи дозволяє змінювати логіку обробки даних, використовуючи фільтрування вищого порядку, алгоритми адаптивної обробки (наприклад, Kalman-фільтр), прогнозування руху або об'єднання з даними GPS для побудови тривимірних траєкторій. Система здатна підтримувати запис даних у локальне сховище (SD-карту) або передавати їх у реальному часі через Bluetooth, BLE або MQTT-протокол до хмарного сервісу для подальшого аналізу. У залежності від сценарію використання програмну

логіку можна адаптувати до періодичного вимірювання (енергоощадний режим), безперервного стрімінгу (реальний час) або накопичення даних у буфері з періодичним вивантаженням.

У практичному аспекті система може бути адаптована до:

– портативних пристроїв для біомеханічного аналізу (моніторинг рухів рук, ніг, постуральної рівноваги);

– робототехнічних систем як модуль зворотного зв'язку положення;

– моніторингових рішень в охороні праці (виявлення падіння, зміна прискорення або кута нахилу);

– спортивних трекерів з виводом параметрів руху у мобільний застосунок або годинник.

Реалізована система не є жорстко орієнтованою на один сценарій використання. Вона має чітко окреслений потенціал адаптації до різних умов за рахунок універсальності обраних компонентів, відкритої програмної екосистеми та модульної побудови. Подальший розвиток проєкту може передбачати створення багатовузлових сенсорних мереж, розподілених систем збору та хмарного аналізу даних, що відкриває перспективи для застосування системи в контексті IoT, розумних пристроїв та мобільного телеметричного спостереження.

ВИСНОВКИ

У результаті виконання всіх поставлених завдань дипломної роботи було реалізовано повний цикл проєктування та дослідження автономної системи збору, обробки та передавання даних з сенсорних модулів на основі мікроконтролера. На першому етапі було здійснено комплексний аналіз технічних вимог, обґрунтовано вибір апаратної платформи (Arduino Nano), сенсорів прискорення та кутових швидкостей, а також модуля Bluetooth Low Energy, що дозволило сформувати оптимальну конфігурацію пристрою з урахуванням обмежень щодо енергоспоживання, габаритів і надійності.

Подальші етапи були присвячені детальному розробленню структурної, принципової та архітектурної схем, алгоритмів збору і передавання даних, а також розробці програмного забезпечення з використанням Arduino IDE та бібліотеки ArduinoBLE. Впроваджені алгоритми обробки сенсорних сигналів із застосуванням експоненційного згладжування (EMA) і комплементарної фільтрації забезпечили стабільність відображення результатів, що підтверджено емпіричними дослідженнями точності та затримки. Було також протестовано стабільність каналу передавання даних у BLE-режимі, що виявило прийнятні значення латентності та втрат пакетів для динамічних задач моніторингу.

Проведено оцінку енергоспоживання системи у різних режимах роботи, що дозволило визначити доцільне джерело автономного живлення та оцінити тривалість автономної експлуатації. Завдяки використанню енергоощадних режимів та оптимізації циклу опитування сенсорів досягнуто балансу між продуктивністю та часом автономної роботи.

Окрему увагу приділено можливостям масштабування, у тому числі шляхом розширення кількості сенсорів, використання альтернативних інтерфейсів зв’язку (Wi-Fi, GSM, LoRa) та адаптації програмної логіки до

конкретних сценаріїв застосування — у медицині, спорті, робототехніці, охороні праці тощо.

Усі поставлені цілі дипломної роботи досягнуто: створено функціональний прототип, який задовольняє вимогам до автономності, надійності, точності вимірювання та стабільності передачі, а також має перспективу подальшого розвитку в рамках концепції IoT або мобільних моніторингових систем.

СПИСКИ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Закон України «Про наукову і науково-технічну діяльність» [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/848-19

2. ДСТУ 2861-94. Пристрої мікропроцесорні. Терміни та визначення.

3. ISO/IEEE 11073 – Health informatics – Personal health device communication – Bluetooth transport profile.

4. Труханов В.М. Основи побудови мікропроцесорних систем керування. – Харків: НТУ "ХПІ", 2021. – 300 с.

5. Сидоренко В.М., Пархоменко І.О. Вбудовані системи: Мікроконтролери та мікропроцесори. – К.: Ліра-К, 2020. – 348 с.

6. Горбачук О.М. Програмовані мікроконтролери Arduino в системах автоматизації. – К.: КНУБА, 2020. – 296 с.

7. Бондаренко П.Ю. Bluetooth технології в бездротових сенсорних мережах. – Львів: Видавництво ЛНУ, 2019. – 198 с.

8. Juraj Kusnir. Programming with ESP32 and Bluetooth. – London: Elektor International Media, 2021. – 224 p.

9. Molloy D. Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux. – Wiley, 2019. – 720 p.

10. Monk S. Programming Arduino: Getting Started with Sketches. – McGraw-Hill Education, 2020. – 192 p.

11. Андрієнко О. В. Мікроконтролерна система з використанням Bluetooth для моніторингу параметрів об’єкта // Вісник НТУУ «КПІ». – 2020. – №3(133). – С. 42–48.

12. Савчук М.О. Безпровідні системи з використанням BLE в задачах моніторингу // Наукові вісті ВНТУ. – 2021. – №4. – С. 55–62.

13. Johnson M., Singh A. Energy-efficient wearable movement tracking system using MPU6050 and BLE // IEEE Sensors Journal. – 2021. – Vol. 21(11). – P. 9851–9860.

14. Al-Fuqaha A. et al. Internet of Things: A Survey on Enabling Technologies, Protocols and Applications // IEEE Communications Surveys & Tutorials. – 2015. – Vol. 17, No. 4. – P. 2347–2376.

15. Chen C., Zhao X. Energy consumption analysis of BLE-based wireless sensor networks // Sensors. – 2020. – Vol. 20(10). – P. 2910.

16. Петренко О.В. Автономна система контролю руху з використанням Arduino // Матеріали Міжнар. конф. «Інформаційні технології 2021». – К., 2021. – С. 118–120.

17. Васильченко І.А. Системи контролю з бездротовими модулями Bluetooth // Тези доповідей XX конференції молодих науковців. – Одеса: ОНАХТ, 2022. – С. 77–80.

18. Li Z., Wang X. Design of a motion measurement system based on MPU-6050 and Bluetooth communication // Proc. of ICME 2022. – IEEE, 2022. – P. 312–316.

19. Arduino Project Hub [Електронний ресурс]. – Режим доступу: https://create.arduino.cc/projecthub

20. Офіційна документація MPU-6050 [Електронний ресурс]. – Режим доступу: https://invensense.tdk.com/products/motion-tracking/6-axis/mpu-6050/

21. Офіційна документація Bluetooth SIG [Електронний ресурс]. – Режим доступу: https://www.bluetooth.com/specifications

22. Espressif Systems: ESP32 Technical Reference Manual [Електронний ресурс]. – Режим доступу: https://www.espressif.com

23. Microsoft Docs: Bluetooth communication in UWP apps [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/uwp/devices-sensors/

24. Іваненко Р.О. Розробка енергозберігаючої системи моніторингу параметрів руху: дис... канд. техн. наук. – ЛНУ, 2022. – 198 с.

25. Степанюк В.М. Архітектура бездротових сенсорних систем з Bluetooth Low Energy: автореф. дис... канд. техн. наук. – ХНУРЕ, 2021. – 160 с.

26. Datasheet HC-05 Bluetooth Module [Електронний ресурс]. – Режим доступу: https://www.electronicwings.com/nodemcu/hc-05-bluetooth-module-interfacing-with-nodemcu

27. Adafruit: Using MPU6050 with Arduino [Електронний ресурс]. – Режим доступу: https://learn.adafruit.com/mpu6050-6-dof-accelerometer-and-gyro

28. SparkFun. Getting Started with IMU Sensors [Електронний ресурс]. – Режим доступу: https://learn.sparkfun.com/tutorials/imu-inertial-measurement-unit-basics

29. Bluetooth Low Energy vs Classic Bluetooth [Електронний ресурс]. – Nordic Semiconductor. – Режим доступу: https://www.nordicsemi.com

30. PlatformIO documentation [Електронний ресурс]. – Режим доступу: https://docs.platformio.org/

ДОДАТОК А

Програмний код розроблювального пристрою

```
#include <Servo.h>
Servo myservo1; //create servo object
Servo myservo2; //create servo object
Servo myservo3; //create servo object

//EMA stands for (Exponential moving average)
//Used pins
//accelerometer
const int x_pin = A0;
const int y_pin = A1;
const int z_pin = A2;
//Potentiometer
const int gx_pin = A3;
const int gy_pin = A4;
const int gz_pin = A5;
//Servo
const int servoPin1 = 3;
const int servoPin2 = 5;
const int servoPin3 = 11;

//variables
//initialization of sensor variable, equibalaent
int sensorValue1 = 0;
int sensorValue2 = 0;
int sensorValue3 = 0;
//EMA
float EMA_a = 0.02;      //initialization of EMA alpha
int EMA_S1 = 0;                  //initialization of EMA S1
int EMA_S1_map = 0;              //initialization of variable for servo control
```

```
int EMA_S2 = 0;                 //initialization of EMA S2
int EMA_S2_map = 0;             //initialization of variable for servo control
int EMA_S3 = 0;                 //initialization of EMA S3
int EMA_S3_map = 0;             //initialization of variable for servo control

int angleX = 0;
int angleY = 0;
int angleZ = 0;

// time
unsigned long startTime = 0;
unsigned long currentTime = 0;
unsigned long elapsedTime = 0;

// Volts per G-Force
const float sensitivity = 0.206;

void setup() {
  Serial.begin(115200);
  analogReference(EXTERNAL);
  EMA_S1 = analogRead(x_pin); //set EMA S for t = 1
  EMA_S2 = analogRead(y_pin); //set EMA S for t = 1
  EMA_S3 = analogRead(z_pin); //set EMA S for t = 1

  myservo1.attach(servoPin1);
  myservo2.attach(servoPin2);
  myservo3.attach(servoPin3);
}
```

```
void loop() {
  currentTime = millis();
  elapsedTime = currentTime - startTime;
  // Read pins and convert to G
  // accelerometer rate
  float ax = (analogRead(x_pin) - 512) * 3.3 / (sensitivity * 1023);
  float ay = (analogRead(y_pin) - 512) * 3.3 / (sensitivity * 1023);
  float az = (analogRead(z_pin) - 512) * 3.3 / (sensitivity * 1023);
  // gyroscope rate
  float gx = (analogRead(gx_pin) - 512) * 3.3 / (sensitivity * 1023);
  float gy = (analogRead(gy_pin) - 512) * 3.3 / (sensitivity * 1023);
  float gz = (analogRead(gz_pin) - 512) * 3.3 / (sensitivity * 1023);

  // serva 1
  move_motor(x_pin, EMA_S1, EMA_S1_map, sensorValue1, gx);
  angleX = complementaryFilter(angleX, gx, elapsedTime, EMA_S1);
  myservo1.write(angleX);//send the latest value to the servo
//  myservo1.write(EMA_S1_map);//send the latest value to the servo

  // serva 2
  move_motor(y_pin, EMA_S2, EMA_S2_map, sensorValue2, gy);
  angleY = complementaryFilter(angleY, gy, elapsedTime, EMA_S2);
  myservo2.write(angleX);//send the latest value to the servo
  //myservo2.write(EMA_S2_map);//send the latest value to the servo

  // serva 3
  move_motor(z_pin, EMA_S3, EMA_S3_map, sensorValue3, gz);
  angleZ = complementaryFilter(angleZ,gz, elapsedTime, EMA_S3);
  myservo3.write(angleZ);//send the latest value to the servo
```

```
 // myservo3.write(EMA_S3_map);//send the latest value to the servo

  delay(100);

  startTime = currentTime;
 }


 void move_motor(int input_pin, int &EMA_S, int &EMA_S_map, int &sensorValue, float gyroVar){
  sensorValue = analogRead(input_pin); //read the sensor
  EMA_S = (EMA_a*sensorValue) + ((1-EMA_a)*EMA_S);//run the EMA
  Serial.print("pin ");
  Serial.print(input_pin);
  Serial.print(": ");
  Serial.print(sensorValue);//the first variable for plotting
  Serial.print(",");//seperator
  Serial.println(EMA_S);//the second variable for plotting including line break
  EMA_S_map = map(EMA_S, 0, 1023, 0, 180);//map ADC values to servo values (0-180)
 };

 //Complementary Filter, similar to exponential, this example is for an accelerometer
 // The 0.98 and 0.02 are the same as the weights in the exponential filter
 //angle = 0.98 *(angle+gyro*dt) + 0.02*acc

 float complementaryFilter(float angle, float gyro, float dt, float acc)
```

```
{
float totalAngle = .5 * (angle + gyro * dt) + .5 * acc;
   return totalAngle;
}
```