Національний університет «Полтавська політехніка імені Юрія Кондратюка»
(повне найменування вищого навчального закладу)
Навчально-науковий інститут інформаційних технологій і робототехніки
(повне найменування інституту, назва факультету (відділення))
Кафедра автоматики, електроніки та телекомунікацій
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до кваліфікаційної роботи

бакалавр
(освітній рівень)

на тему **Розробка самохідної, телеметричної, радіокерованої системи**

Виконав: студент 4 курсу, групи 401-тт
спеціальності 172 «Телекомунікації та радіотехніка»
(шифр і назва напряму підготовки, спеціальності

Карпук В.Ю.
(прізвище та ініціали)
Керівник Обіход Я.Я.
(прізвище та ініціали)
Рецензент Шефер О.В.
(прізвище та ініціали)

Полтава - 2021 рік

РЕФЕРАТ

кваліфікаційної роботи " Розробка самохідної, телеметричної, радіокерованої системи "

Робота містить 64 сторінок, 30 ілюстрацій, 19 таблиць, 28 використаних джерел.

Ключові слова: ESP32, Arduino, мікроконтролер, дистанційне керування, телеметрія.

Об’єктом розроблення кваліфікаційної роботи є – дистанційно керована платформа з можливостями телеметрії.

Предметом кваліфікаційної роботи є – основні принципи побудови дистанційно керованих телеметричних систем.

Метою даної кваліфікаційної роботи є:

- Проведення аналізу минулих та сучасних дистанційно керованих систем;
- Розбір загальних принципів побудови віддалено керованих систем;
- Аналіз сучасного асортименту компонентів для розробки дистанційних систем.
- Розробка проекту самохідної, телеметричної, радіокерованої платформи.
- Розробка додатку керування радіокерованою платформою.

В ході даного дипломного проекту буде проведено аналіз сучасних роботизованих систем та опираючись на отримані знання, створено мобільну, телеметричну платформу із дистанційним управлінням по Wi-Fi та додаток керування для неї.

ABSTRACT

qualification work "Development of self-propelled, telemetry, radio-controlled system"

The work contains 64 pages, 30 illustrations, 19 tables, 28 used sources.

Key words: ESP 32, Arduino, microcontroller, remote control, telemetry.

The object of the development work is qualifying - remotely operated platform with the capabilities of telemetry.

The subject of qualifying work is fundamental principles of construction remotely controlled telemetry systems.

The purpose of this qualification work is:

- Analysis of old and modern remotely controlled systems;
- Analysis of general principles of building remotely controlled systems;
- Analysis of the modern assortment of components for the development of remote systems.
- Development of the project of the self-propelled, telemetric, radio-controlled platform.
- Development of a radio-controlled platform control application.

In the course of this diploma project the analysis of modern robotic systems will relying on the acquired knowledge, will be created a mobile, telemetry platform with remote control via Wi - Fi and a control application for it.

Національний університет «Полтавська політехніка імені Юрія Кондратюка»
Інститут Навчально-науковий інститут інформаційних технологій та робототехніки
Кафедра Автоматики, електроніки та телекомунікацій
Освітньо-кваліфікаційний рівень Бакалавр
Спеціальність 172 «Телекомунікації та радіотехніка»

ЗАТВЕРДЖУЮ

Завідувач кафедри автоматики, електроніки та телекомунікацій

_____О.В. Шефер
“ 11 ” травня 2021 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Карпуку Владиславу Юрійовичу

1. Тема роботи «Розробка самохідної, телеметричної, радіокерованої системи»
керівник роботи Обіход Ярослав Якович, к.т.н., доцент
затверджена наказом вищого навчального закладу від 03.03.2021 року № 158 -фа

2. Строк подання студентом проекту (роботи) 15.06.2021 р.

3. Вихідні дані до проекту (роботи) мікросхема ESP32; Arduino Nano, давачі телеметричної системи: DHT-11, US-025, A3144, MPU-6050.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити). Обґрунтування необхідності розробки самохідної, телеметричної, радіокерованої системи. Розрахунок основних параметрів платформи. Вибір апаратного забезпечення при побудові дистанційно керованої платформи.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових плакатів):
   1) Функціональна схема побудови радіокерованої моделі;
   2) Будова сучасного мікроконтролеру;
   3) Схема підключення інтерфейсу UART;
   4) Принципова схема платформи.

6. Дата видачі завдання 11.05.2021 р.

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів дипломної роботи	Термін виконання етапів роботи			Примітка (плакати)
1	Теорія та загальні відомості про дистанційно керовану техніку.	18.05.21			
2	Апаратне забезпечення та його вибір при побудові дистанційно керованої платформи.	02.06.21			
3	Створення платформи, додатку керування та розрахунки їх характеристик.	09.06.21			
4	Оформлення кваліфікаційної роботи бакалавра.	15.06.21	II		

Студент _____ Карпук В.Ю.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Обіход Я.Я.
(підпис) (прізвище та ініціали)

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

СТРП – самохідна, телеметрична, радіокерована платформа

БПА – безпілотний апарат

ОС – операційна система

ЗМІСТ

ВСТУП

Майже будь-якій галузі не будуть зайвими ті переваги, які надає впровадження робото-технічних систем і для них потрібні системи контролю, керування, моніторингу та спеціалісти, які обслуговують їх. Що б краще розумітися в цій темі потрібно розглянути такі поняття, як: робототехніка і дистанційне керування та їх класифікація.

Телеметричне керування – це такий вид управління на відстані при якому передаються як команди управління, так і данні із датчиків, можлива також передача відео сигналу. Це завжди не пряме, віддалене керування, про те це не значить, що оператор знаходиться далеко, він може бути и за декілька метрів, як наприклад хірургічний маніпулятор і навпаки може бути на великій відстані, як оператор наукового підводного судна [1].

Робототехніка – це суміш інженерії із інформатикою вона займається конструюванням, проектуванням та експлуатацією роботів для автоматизації робочих або наукових процесів [2].

Роботи за типом дистанційної роботи можна поділити на три групи [2, 3] :

- **Автономні:** техніка що здатна працювати без команд з боку людини, або потребує її мінімального втручання, зазвичай на етапах встановлення та впровадження робота у роботу.
- **Ручні:** такі машини не здатні працювати самостійно і потребують команд керування у реальному часі від оператора.
- **Напівавтомат:** таких роботів не можна назвати ні ручними, ні автономними. Вони отримують дані із датчиків та можуть на них реагувати, про те вони також можуть отримувати команди від людини оператора. автоматичне так і ручне.

Із розвитком технологій людство змогло створити дистанційне керування для роботів та різних електронних систем.

Дистанційне керування буває декількох видів [2, 3]:

- **Світлове випромінювання** – керування здійснюється за допомогою посилання сигналів інфрачервоним чи видимим світлом.
- **Радіо керування** – у якості носія сигналу використовуються радіохвилі різних частотних діапазонів.
- **Технологія розпізнавання мови** – таке управління дозволяє керувати процесами у звичній людині формі – усній.
- **Акустичне керування** – із назви може бути схожим на технологію розпізнавання мови, проте це не так.
- **Мережеве керування** – це вид управління за допомогою різних мережевих технологій таких як: Wi-Fi, Bluetooth, Zigbee, Z-Wave.

Актуальність даної роботи обумовлена все більшим розповсюдженням автоматичних систем – у науковій, воєнній, промисловій та медичних сферах. Такі автоматичні системи (роботи) можуть містити в собі виконуючі прилади для взаємодії із навколишнім середовищем такі, як маніпулятори, сопла для зварювання, крашення, склеювання, двигуни із колісними, гусеничними чи гвинтовими рушіями. Для зворотного зв’язку вони мають набір датчиків та камер тому можуть відповідно реагувати на середовище навколо них. У наш час існує багато роботів як мобільних так і стаціонарних з дистанційним керуванням так і повністю автономних. Таким чином дослідження теми розробки дистанційного керованої самохідної платформи є актуальним на сьогоднішній день.

Новизною даної роботи є те, що – в ході проведення аналізу теоретичних та практичних даних в існуючих системах дистанційного керованих апаратів було виявлено, що попри їх доступності широкій публіці – відсутній зручний крос-платформовий спосіб керування та база для легкого встановлення та налаштування майже будь-яких сенсорів. З метою покращення розробки та експлуатації подібних систем було запропоновано створення такого крос-платформового додатку для керування та зняття показників датчиків на базі самохідної радіокерованої платформи.

1. ТЕОРІЯ ТА ЗАГАЛЬНІ ВІДОМОСТІ ПРО ДИСТАНЦІЙНО КЕРОВАНУ ТЕХНІКУ

1.1 Історія робототехніки та радіокерованих моделей

1.1.1 Дистанційно керовані ігрові моделі

Великий розвиток керованих роботів дали саме моделі дистанційно керованих іграшок, тому потрібно розглянути цей виток історії детальніше.

Перші радіокеровані іграшки стали можливі завдяки розвитку калільного двигуна, який при малих розмірах надавав достатню для малих моделей потужність. Проте керування на той момент було лише у вигляді увімкнути та вимкнути двигун, розвиток повноцінного керування із поворотом передніх коліс та потужністю двигуна, з'явився лише в 60-х роках.

В 1784 р. британський винахідник Вільям Мердок працюючи над покращенням парового двигуна запропонував модель самохідної карети на паровому ходу, що здатна перевозити до 5 т – це була одна із самих перших машин здатної до самостійного руху, проте поки що без дистанційного керування [4].

1.1.2 Дистанційно керована техніка у воєнній сфері

Воєнні конфлікти в історії людства – це двигуни прогресу, тому потрібно детальніше розібрати технології які використовувались в той час.

В 1930-х роках радянський союз провів іспити на танку МС-1 із встановленим телеуправлінням.

В ході іспитів було визначено доцільність розробки подібних радіокерованих танків. У 1933 р. з'явився спеціалізований танк ТТ-18, в якому все керування було дистанційне. Система управління танка могла вмістити 16 команд, із них були: зміна швидкості, повороти корпусом, зупинка двигуна та підрив вибухівки. Реальна максимальна дальність передачі команд становила 500-1000 м. Танк в цілому виявився успішним проектом (як радіотанк) він мав легке керування та гарну прохідність, про те із за конструктивних особливостей бази танка він повертався при їзді на вибоїнах, тому в серію не пішов.

В 1940 р. на базі танка БТ-7 розробили новий телетанк, він уже мав автономність 4-6 годин, дальність керування до 4 км та 17 командну систему управління. Саме керування видалось вдалим, проте як бойова машина він мав погану точність стрільби тому від нього відмовилися [5].

1.1.3 Роботи для екстремальних умов

Також радянськими вченими після катастрофічної події на ЧАЕС 1986 р. було розроблено мобільні роботи для проведення в зоні робіт. Загалом було приблизно 15 роботів, серед них були моделі такі як: гусеничний «Мобот-ЧХВ», робот «СТР-1» та колісний робот розвідник «РР-1».

Легкі роботи проводили заміри радіаційної обстановки та передавали телеметричні дані для візуалізації та контролю робочої зони інших роботів. Більш важкі роботи були призначення для проведення технологічних робіт із прибирання та дезактивації майбутніх будівничих площадок.

Загалом роботи заміняли собою роботу сотень людей не наражаючи їх на небезпеку, адже місцями рівень радіації складав 18 000 Р/г [6, 7]. Приклад одного із таких роботів можна бачити на рисунку (рис. 1.1) .

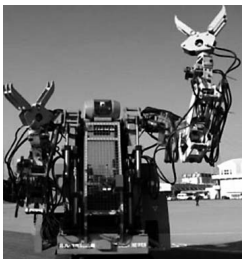


Рисунок 1.1 – Японський протиаварійний мобільний робот Т-52

1.2 Аналіз сучасного стану робототехніки та його перспективи

Сучасних роботів із дистанційним керуванням можна розділити за наступним типом використання [3, 8]:

- Космічні апарати, більшість з них має на борту можливості дистанційного керування.
- Роботи воєнного призначення, широко використовуються для розвідки та нанесення ударів по противнику.

- Дистанційні маніпулятори для віддаленої роботи оператором із небезпечними матеріалами, наприклад радіоактивними чи вибухонебезпечними.
- Техніка наукового призначення, включає в себе, як планетоходи на поверхні інших планет так і підводні човни для різноманітних досліджень, де перебування людини не можливе або економічно не доцільне.
- Промислові маніпулятори для різних робіт – зварювання, вантаження, різка, фарбування.

1.2.1 Космічні програми використання дистанційної техніки

Відкритий космос – це величезний простір для експлуатації роботів із дистанційним керуванням, адже находження там людини економічно не доцільне, а у більшості випадків, ще і небезпечне чи взагалі не можливе.

Космічні об’єкті із дистанційним керуванням у космосі можна поділити на наступні групи:

- Штучні супутники – це спеціалізовані пристрої, що використовуються для ретрансляції сигналів, прогнозів погоди чи роботи систем позиціювання (GPS, ГЛОНАСС).
- Наукові роботи із вивчення космосу – це різноманітні космічні станції, розвідувальні зонди, спостерігачі, десантні роботи, всі вони потребують розвинених систем віддаленого управління.
- Воєнні космічні об’єкти – про них відомостей не дуже багато, про те відомо, що вони дистанційно керовані.

Оскільки найбільш відомими віддалено керованими об’єктами в сонячній системі є апарати створені по космічній програмі НАСА «Mars Exploration Rover» (MER), тому доцільно розглянути їх детальніше [9].

В ході програми MER було створено декілька марсоходів, тому розглянемо їх загальні характерні для них особливості.

Цілі які ставились перед цими апаратами були:

- вивчення марсіанських ґрунтів та порід, тобто використовувались маніпулятори для збору зразків та системи їх аналізу.
- Пошук цінних залізоносних мінералів, для цього використовували камери із різними світлофільтрами.
- Пошук води та мінералів які могли свідчити про її наявність.
- Оцінка якості життя в майбутньому на Марсі.

Сам марсохід представляв собою апарат на колісній базі, масою близько 200 кг, декілька камер, спектрометри, маніпулятор із вбудованими датчиками та мікро-камерою.

Марсіанські планетоходи являються гібридними дистанційно керованими апаратами. Вони приймають команду на рух від оператора, але як саме рухатися по якій траєкторії вони вирішують самі, завдяки побудові стереоскопічної карти навколишнього середовища, з якої вони розумують де перешкоди та оминають їх. Вони аналізують перешкоди по критеріям висоти перешкоди, щільності ґрунту, куту нахилу поверхні та із десятків можливих шляхів обирають найбільш безпечний та коротший шлях. Планетохід можна побачити на рисунку нижче (рис. 1.2) [10].

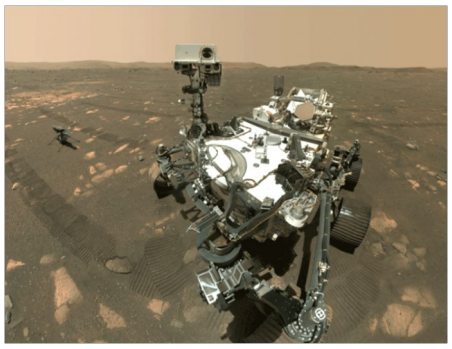


Рисунок 1.2 – Марсохід «Perseverance»

1.2.2 Воєнні розробки у сфері дистанційно керованих апаратів

Як і в космічній галузі, військові програми потребують розвідувальної техніки, оскільки інформація про ворога є дуже цінною. Розвідувальні дистанційно керовані апарати найбільш розповсюджені у військовій справі.

Яскравим прикладом такого є БЛА – безпілотний літальний апарат. Вони можуть бути різного ступеню автономності, від керованих в ручному режимі оператором, до майже повністю автоматичних. Загалом їх основа перевага це суттєво менша вартість порівняно із традиційним літаком та відсутність ризику людських втрат [11].

Систему БЛА складає:

- Сам безпілотний літак;
- Система зв'язку: це може бути радіо чи супутниковий зв'язок;
- Пункт управління оператором;
- Додаткове обладнання необхідне для транспортування чи обслуговування БПА.

Особливий тип техніки воєнного призначення є роботи-сапери, вони виконують розмінування вибухонебезпечних пристроїв. Зазвичай структурно вони складаються із гусеничної платформи, декількох камер та маніпуляторів, можуть також бути обладнані водо/піно пушками.

Яскравим прикладом такої техніки є БПА «Фантом» (рис. 1.3).

Рисунок 1.3 – Український багатоцільовий тактичний апарат «Фантом»

1.2.3 Підводні човни дистанційного керування

У випадку дистанційно керованих апаратів-човнів (remotely operated vehicles, ROV), більш економічно доцільного та безпечного шляху виконання підводних маніпуляцій не має.

ROV апарати значно відрізняються за розмірами. Від таких, що вміщають у себе лише телекамеру і використовуються лише для спостерігання, до систем котрі мають механічні прибори, декілька гнучких маніпуляторів та інше обладнання.

Самим розповсюдженим класом є робочі апарати, що мають досить велику потужність (50-100 к. с.). Вони можуть перевозити до 300 кг вантажу, деякі навіть до 500кг. Типовими задачами такого класу апаратів є легке підводне будівництво, дослідження труб або бурові роботи [12].

На сьогоднішній день такі апарати можуть виконувати цілий спектр задач такі як: наукові дослідження, інженерні перевезення чи встановлення, пошуково-рятувальні операції.

1.2.4 Інженерна дистанційна техніка

Беручи до уваги те, що загальний рівень технологій вже дозволяє будувати майже автономні самохідні платформи, то цим не змогли не скористатися у інженерній справі. Сама техніка являє собою транспортний засіб на якому встановлені відповідні інструменти, прибори та сенсори для виконання циклу технологічних операцій. Мобільні інженерні платформи виконують ряд робіт у нафтових, гірничих, газових та цивільному будівництві областях.

Основними цілями інженерних мобільних платформ є [13]:

- Зменшення кількості персоналу;
- Проведення робіт в умовах шкідливих чи небезпечних для людини;
- Збільшення мобільності та ефективності при проведення гірничих, земляних та інших робіт;
- Зменшення у робітників втоми та робітничого травматизму.

Сучасні застосування мобільних платформ це – пошуково-рятувальні роботи при стихійних лихах, шахтові роботи із видобутку ресурсів, лісозаготовчі роботи і т.д. Прикладом такої техніки є інженерний робототехнічний комплекс РТС-У (рис. 1.4).

Рисунок 1.4 – Інженерний комплекс РТС-У

1.2.5 Медична техніка віддаленого керування

Завдяки розвитку роботизованої техніки, почалась революція у медичній галузі. Так наприклад розвиток пристроїв візуалізації дав змогу ставити більш точні діагнози та проводи менш інвазійні методи лікування. Загалом під тему дистанційно керованих пристроїв підходить роботизованої хірургії, область із великими перспективами.

Роботизована хірургія була створена із ціллю розширити можливості хірургів, які виконують операції [14]. Хірург має можливість проводити операцію двома методами:

- Власноруч керуючи дистанційним маніпулятором із камерами, представляючи собою роботизовані руки;
- За допомогою комп'ютерного управління, де хірург лише віддає команди як саме провести операцію, цей метод гарний тим, що може не потребувати присутності хірурга.

Завдяки досягнення роботизованої хірургії, операції проводяться з великою точністю, меншою кількістю втручання в організм людини та мають менші крововтрати. Крім цього хірурги мають більший контроль над зоною операції, менше втомлюються оскільки їм не потрібно весь час стояти над пацієнтом, також природне тремтіння пальців фільтрується програмним забезпеченням роботу. Популярним рішенням такого дистанційного медичного обладнання є роботизована хірургічна система «da Vinci» (рис. 1.5).

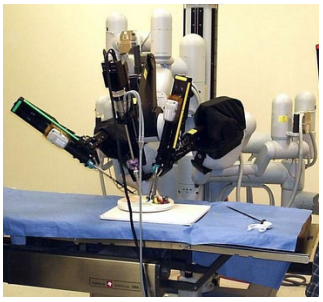


Рисунок 1.5 – Хірургічна система «da Vinci»

1.3 Загальні принципи побудови радіокерованої моделі

1.3.1 Основні компоненти

Для ефективної побудови віддалено керованих роботів потрібно розібратися у базових компонентах із яких вони складаються. Загалом у їх склад входять такі компоненти як: корпус, шасі, блок живлення, датчики, виконавчий механізм, блок керування, приймально-передавальний пристрій та програмне забезпечення [15].

Більш наглядно компоненти радіокерованої моделі та їх взаємодію можна побачити на малюнку нижче (рис 1.6).

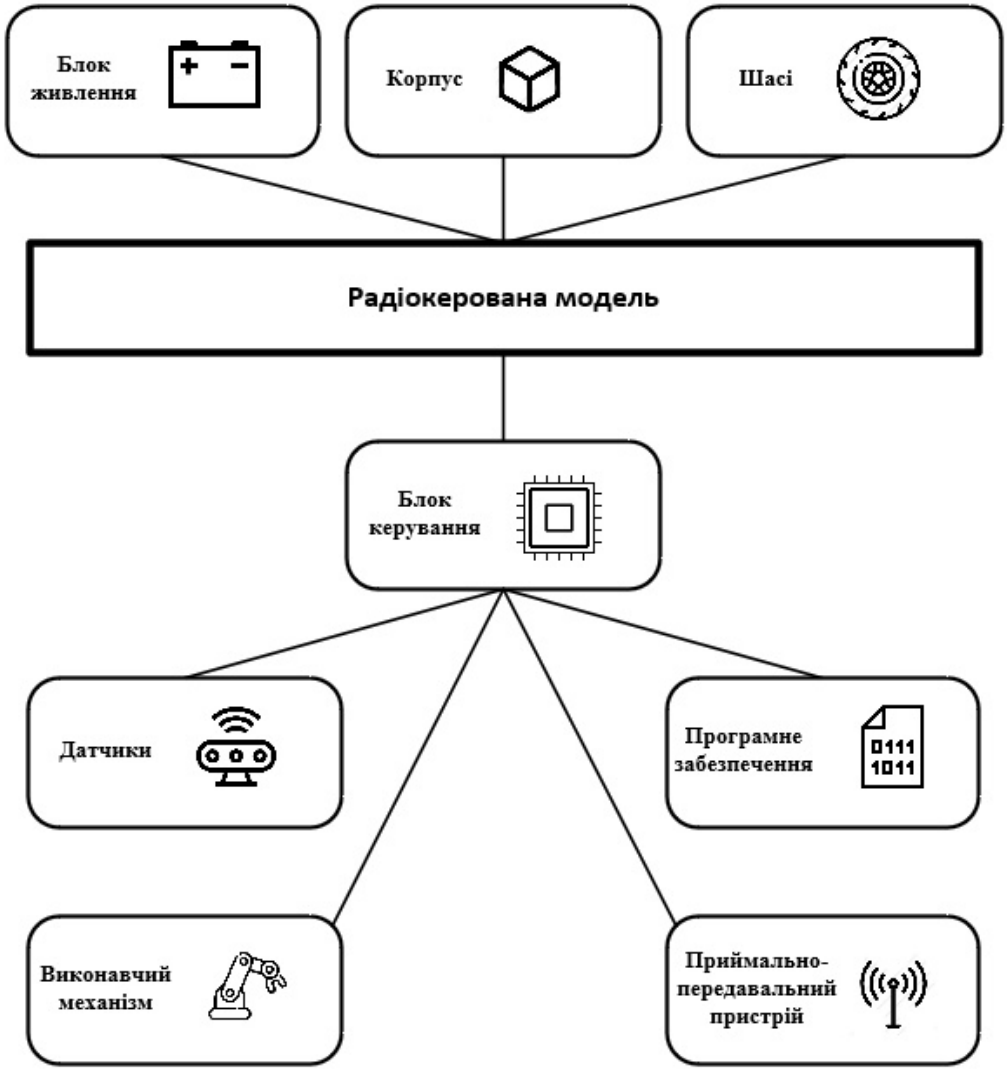


Рисунок 1.6 – Функціональна схема побудови радіокерованої моделі

1.3.2 Корпус

Основою будь-якої автономної, дистанційно керованої техніки є – корпус. Саме він надає базу для подальшої побудови робота. Від його вибору залежить те які характеристиками буде мати та чи інша система [16].

Для корпусу можуть використовуватися такі матеріали:

- Алюміній – легкий метал, що надає міцність та достатньо стійкий до корозії.
- Акрил – має популярність через свій гарний прозорий вигляд.
- Фанера – доволі важка та може мати проблеми із за вологості, проте дуже легко обробляється.
- Полікапролактон – пластик, що має низьку температуру плавлення, використовується зазвичай для прототипування.
- Різного роду пластмаси – золота середина між міцністю, вагою та складністю обробки.
- Сталь – надає величезну міцність та зносостійкість, проте має велику вагу, використовується у техніці важливого призначення.

1.3.3 Шасі

Шасі будь-якої мобільної платформи, більше ніж інші компоненти відповідає за її тип: колісна техніка, гусенична чи можливо якісь екзотичні рушії типу шнекових [23]. Далі буде розглянуто популярні на даний час варіанти шасі.

Колісні – дуже розповсюджений варіант шасі для мобільних систем, відзначаються своєю простотою створення та розробки, мають середню прохідність, найшвидші із наземних варіантів шасі. Із недоліків є радіус повороту, проте в деяких спеціальних колесах такий недолік відсутній (колесо Ілона).

Крокуючі – це різноманітні конструкції, які згинаються на шарнірах. Пересування здійснюється за рахунок синхронного переміщення опорних конструкцій. Перевагою є досить велика прохідність. Недоліки: складність розробки та створення і не тільки конструкції, а ще і програмного забезпечення.

Гусеничні – мають велику прохідність, малий радіус повороту, можливість повороту на місці, велику стійкість до перевертань. Недоліки: складне створення конструкції, особливо, якщо наявна підвіска, потребують частого технічного огляду.

Гібридні – це такий вид шасі, що може в себе включати усі різновиди, наприклад гусениці позаду машини та колеса спереду. Вони поєднують недоліки та переваги усіх видів шасі, що входять до їх складу, тому огляд таких шасі можливий лише на конкретному прикладі.

1.3.4 Блок живлення

Блок живлення – це такий функціональний блок самохідної системи, що надає електроенергію іншим блокам машини. Він може бути відсутнім або замінений на адаптер, якщо живлення подається по кабелю, таке часто буває у підводних чи воєнних самохідних системах. Найбільш популярним є рішення коли в якості блока живлення виступають батареї чи акумулятори. Від блока живлення залежить потужність двигунів, а отже прохідність самохідної системи, але в першу чергу блок живлення це – час автономності всієї системи в цілому.

1.3.5 Датчики

Датчик або сенсор – це такий прилад, головне призначення якого переводити вимірювання фізичних величин навколо самохідної системи у електричні сигнали для подальшого опрацювання.

Тобто датчики це місток між навколишнім середовищем та роботом [18].

Датчиків існує дуже велика кількість, тому перерахуємо основні дані які можна отримувати за допомогою них:

- Рівень освітлення – фотодіоди;
- Вимірювання відстаней – лазерні, ультразвукові далекоміри;
- Рівень прискорення по декільком осям – акселерометри;
- Рівень нахилу – гіроскопи;
- Рівень вологості/дощу – резистивні датчики;

- Рівень звуку – мікрофони;
- Рівень магнітного поля – магнітометри;
- Рівень температури та вологості – термометри та гігрометр;
- Рівень напруги та струму – аналого-цифрові перетворювачі (АЦП);
- Складні модульні датчики – GPS, датчики лінії, жестів, голосових команд, іонізуючого випромінювання і т.д.

Деякі із датчиків являють собою модулі, які дозволяють проводити вимірювання відразу декількох фізичних величин та обробляють вихідні дані, що дозволяє із одних величин отримувати інші у аналоговому чи цифровому вигляді.

1.3.6 Виконавчий механізм

Виконавчий механізм – це той пристрій яким самохідна система взаємодіє із навколишнім середовищем, це може бути ліхтар або маніпулятор для переміщення предметів. Зазвичай в складі такого механізму присутні електродвигуни, проте бувають і інші приводи наприклад пневмоприводи [19].

1.3.7 Блок керування

Блок керування представляє собою одну чи декілька електронних плат, що можуть містити мікроконтролери, вони займаються обробкою інформації, що поступає із датчиків та приймає відповідні рішення на основі них, наприклад приводить у дію виконуючі механізми.

Крім цього блок керування отримує віддалені команди від оператора та може керувати іншими функціональними блоками самохідної системи. Усього може бути декілька блоків керування кожен із яких виконує певну функції чи введений для резервування, в такому випадку усі вони об’єднуються під керівництвом центрального блока керування.

1.3.8 Приймально-передавальний пристрій

Приймально-передавальний пристрій – являє собою пристрій, що може приймати та передавати повідомлення по навколишньому середовищу, це може бути радіохвилі, лазерне чи світлове випромінювання, або електромагнітні хвилі по кабелю. Саме від його характеристик залежить дальність та швидкість передачі інформації, тобто можливість деяких сценаріїв використання мобільної дистанційної керованої системи. Його призначення у складі самохідної системи – це передавати телеметричну інформацію яку отримує система від датчиків та отримувати керуючі сигнали від оператора.

1.3.9 Програмне забезпечення

Програмне забезпечення (ПЗ) – це керуюча програма чи програми для самохідної системи, в яку входять алгоритми обробки даних із датчиків, їх калібрування, аналіз, перетворення для зручної форми. Також у ПЗ описані алгоритми обробки команд від оператора, що б виконавчі механізми працювали так як очікується.

У сучасному світі та розвитку технологій, програмне забезпечення є однім із найціннішим із компонентів самохідної телеметричної системи, адже від програми залежить, як саме вона буде виконувати поставлені перед собою завдання.

ВИСНОВОК

У ході першого розділу було розглянуто побудову дистанційно керованих систем на прикладі іграшкових моделей, військової та інженерної техніки було проведено аналіз сучасного стану робототехніки, а саме дистанційно керованих систем та проведено їх класифікацію.

Під час класифікації було визначено основні сфери дистанційно керованої техніки, це космічні апарати, військова, наукова техніка та маніпулятори для промислових потреб, окрім цього також є медична техніка із телеметричним керуванням. Усі класи даної техніки мають певні специфічні для свого використання вимоги і вони дуже різняться в залежності від призначення техніки.

Оглянувши історію та сучасний стан робототехніки було сформовано основні принципи побудови самохідної телеметричної радіокерованої платформи. Було визначено принципи конструювання компонентів СТРП таких як: корпус, шасі, блок живлення, блок датчиків, виконавчий механізм, блок керування та приймально передавальний пристрій.

Завдяки аналізу класифікації та принципів побудови дистанційно керованих систем можливо розпочати роботу над пошуком апаратного забезпечення у наступному розділі.

2. АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ ТА ЙОГО ВИБІР ПРИ ПОБУДОВІ ДИСТАНЦІЙНО КЕРОВАНОЇ ПЛАТФОРМИ

2.1 Опис самохідної, телеметричної, радіокерованої платформи

Самохідна, телеметрична, радіокерована платформа (СТРП) – представляє собою апаратно-програмний комплекс, який складається із самої самохідної платформи та терміналу керування.

СТРП повинна виконувати наступні завдання:

- Мати змогу самостійно переміщуватися у просторі;
- Збирати та обробляти у зручну форму дані отримані від різних датчиків, які можуть бути встановлені на платформі;
- Підтримувати зв'язок із терміналом керування через Wi-Fi мережу;
- Відправляти отримані дані від датчиків на термінал керування;
- Дані, які повинна збирати СТРП – температура, вологість, відстань до перешкоди, власна швидкість, атмосферний тиск, орієнтація у просторі.
- Управління оператором здійснюється через термінал керування;
- Потокова трансляція FPV відео на термінал керування;
- Термінал керування повинен забезпечити інтерфейс доступу інших можливих сервісів до СТРП.

Керування СТРП відбувається дистанційно через термінал керування. Термінал керування представляє собою звичайний смартфон на операційній системі (ОС) "Android" із встановленим на ньому додатком керування СТРП. Вибір саме такого методу управління обумовлюється простотою встановлення додатку та широкою розповсюдженістю смартфонів із ОС "Android" (75% всіх смартфонів на ринку), тобто термінал керування дуже просто в разі чого зробити із більшості смартфонів методом інсталяції додатку керування СТРП.

Дистанційно керована телеметрична платформа представляє собою ходову базу із гусеничними рушіями. Конструктивно платформу можна поділити на дві частини внутрішню і зовнішню.

Для кращого наочного сприйняття опишемо будову СТРП.

У внутрішньому корпусі розташовуються:

- два літієвих акумулятори формфактору 18650;
- два електричних колекторних двигуна із редукторами, по одному на кожну гусеницю;
- DC-DC перетворювач напруги;

Ззовні знаходяться:

- Плата 2-х МП камери та Wi-Fi модуль;
- Набір датчиків: термометр, гігрометр, ультразвуковий далекомір, барометр, одометр, спідометр, акселерометр та гіроскоп.
- Плата Arduino Nano із адаптером до неї;
- Комутаційна плата;
- Плата заряду Li-ion акумуляторів;
- П'єзокерамічний випромінювач для сповіщення;
- Вимикач.

Даний проект слід розглядати, як макет з метою показати можливості та перспективи повноцінної самохідної платформи. Вибір пав саме на гусеничні рушії, оскільки серед сухопутних рушіїв вони мають найбільшу прохідність та дають можливість дуже точного управління, наприклад розвороти на місці, що є дуже доречним у випадку використання роботизованої техніки. При подальшому розвитку даної платформи можливе її використання в таких галузях, як наукові дослідження, пошуково-рятівні роботи чи воєнний розвідник, оскільки її базу можливо досить легко масштабувати, а датчики замінювати в разі необхідності.

2.2 Асортимент можливого обладнання для розробки платформи

У цьому підрозділі будуть розглянуті сучасні наявні на ринку компоненти для конструювання самохідних радіокерованих систем. Перевагу у детальності розгляду буде віддано мікроконтролерам та датчикам, оскільки саме вони і надають можливості виконувати поставлені перед самохідною платформою завдання.

З огляду на те, що оглянути увесь асортимент можливого обладнання, що наявний на ринку не можливо, будуть розглянуті найбільш типові та популярні рішення, які використовуються для побудови подібних самохідних платформ.

Деяке обладнання може мати у своєму складі ознаки, як мікроконтролерів так і датчиків, тому однозначно буде важко їх класифікувати, але для спрощення будемо відносити їх до тієї чи іншої категорії.

2.2.1 Мікроконтролери

Мікроконтролер являє собою плату чип на якому розміщені, мікропроцесор, постійна та оперативну пам’ять, користувацькі порти вводу-виводу та інші допоміжні блоки такі, як: лічильники, АЦП, та іншу периферію, яка може змінюватися в залежності від виробника обладнання [20] (рис. 2.1).

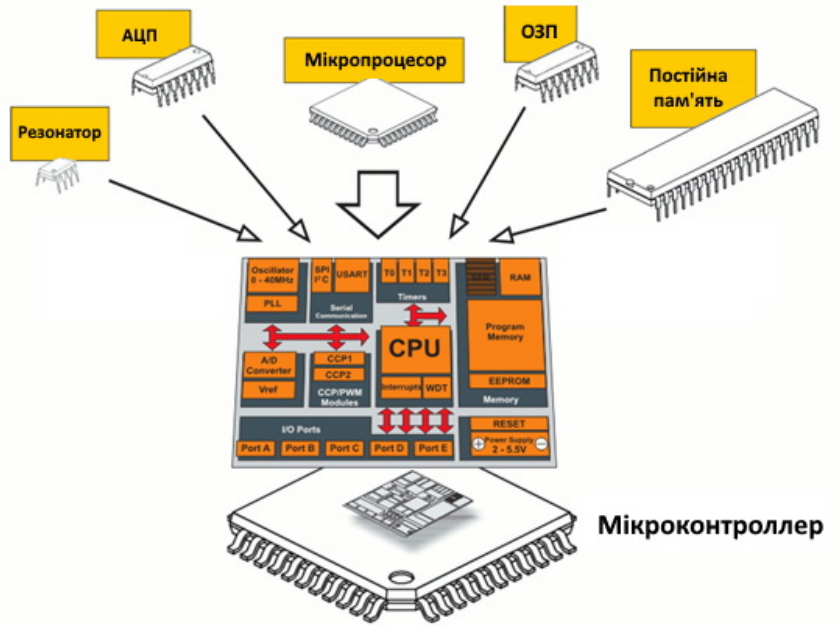


Рисунок 2.1 – Будова сучасного мікроконтролеру

Мікроконтролери бувають у вигляді просто чипів або готових плат розробника, на якій уже розміщені: сам мікроконтролер, струмопровідні доріжки, кварцові резонатори, резистори підтяжки, стабілізатори напруги, тобто усе, що може знадобиться при роботі із мікроконтролером. Використовувати у реальній роботі мікроконтролер у вигляді чипу без нічого не можливо, тому будемо оглядати плати розробника. Звичайно готові плати мають переваги у вигляді уже готових плат із вже розпаяними на них компонентами, але їхня універсальність є також і недоліком, так як не в усіх проектах необхідно мати ті чи інші компоненти, проте оскільки в цій роботі ми створюємо прототип, ці плати добре нам підходять. Приклад такої плати можна побачити на рисунку (рис. 2.2)

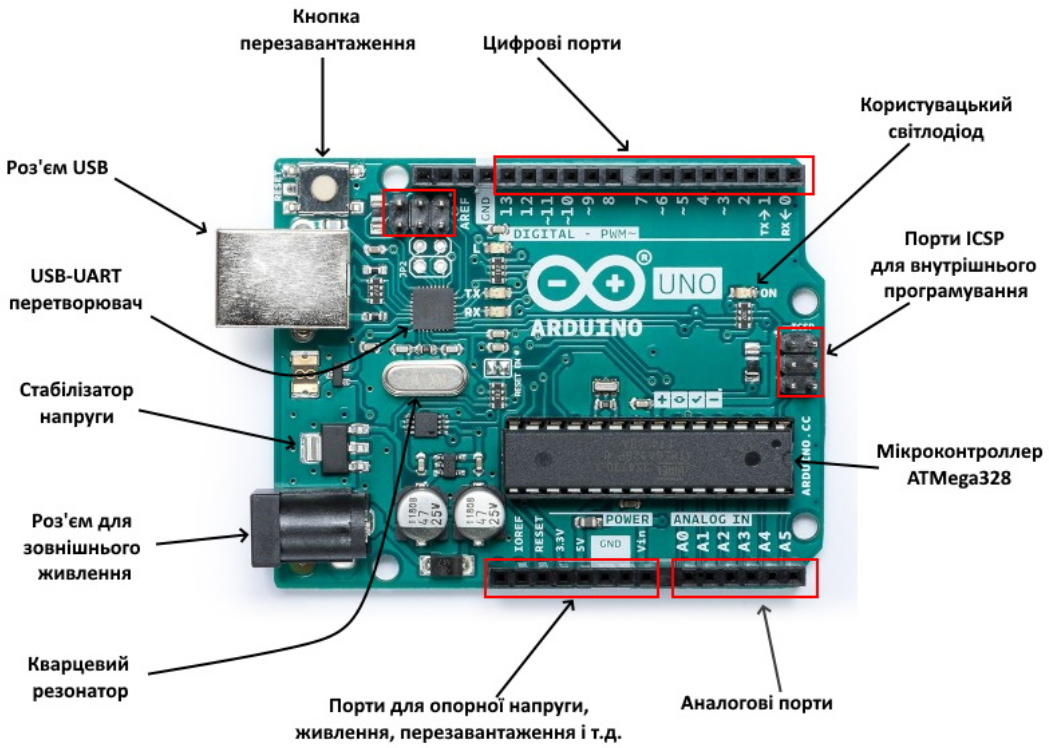


Рисунок 2.2 – Приклад плати розробника «Arduino UNO»

У складі самохідної телеметричної системи плата із мікроконтролером логічно з'єднуючи між собою сенсори, механізми та інші компоненти – виконує функцію блоку керування.

Розглянемо типові рішення на мікроконтролерах для побудови самохідної, телеметричної радіокерованої системи.

Arduino UNO/NANO – ці дві плати (nano та uno) майже нічим окрім форм-фактору не відрізняються тому будемо розглядати їх разом. Це маленькі самодостатні плати, nano від uno відрізняється меншими розмірами та відсутністю роз'єму зовнішнього живлення (можливість прямого підключення є).Наявний на платі конвертер USB-UART забезпечує зв'язок плати із комп'ютером. Ці плати дуже популярні на них дуже багато документації та різних інструкцій.

Приведемо їх основні характеристики (табл. 2.1):

Таблиця 2.1 – Характеристики Arduino UNO/NANO v3

Компонент	**Характеристики**
Мікроконтролер	ATmega328
Тактова частота [МГц]	16
Об'єм FLASH [кБ]	32
Об'єм SRAM [кБ]	2
Об'єм EEPROM [кБ]	1
Цифрові порти	14
Портів АЦП	8
Портів ШІМ	6
Максимальний струм на порт [мА]	40
Інтерфейси	USB; SPI; I2C; UART
Розрядність АЦП [біт]	10
Розрядність ШІМ [біт]	8
Номінальна напруга [В]	5
Розміри [мм]	690*540; 185*42

NodeMCU ESP8266 – це плата розробника, основою якого є мікроконтролер ESP8266. Взагалі цей чип створювався для інтернету речей (IoT), але на даній платі є USB підключення, стабілізатори напруги та розведені виводи, що дозволяє цю плату використовувати в якості плати розробника. Плата має досить швидкий процесор, проте частина його ресурсів йде на підтримку Wi-Fi.

Із головних особливостей плати потрібно відзначити це – модуль Wi-Fi, можливість програмування за допомогою мови Lua або C++ та сумісність плати із Arduino датчиками та модулями.

Приведемо основні характеристики (табл. 2.2):

Таблиця 2.2 – Характеристики NodeMCU ESP8266

Компонент	**Характеристики**
Мікроконтролер	ESP8266
Тактова частота [МГц]	80
Wi-Fi	2.4 ГГц 802.11 b/g/n
Об’єм FLASH [кБ]	4096
Об’єм SRAM [кБ]	80
Об’єм EEPROM [кБ]	-
Цифрові порти	11
Портів АЦП	1
Портів ШІМ	4
Максимальний струм на порт [мА]	12
Інтерфейси	USB; UART; SDIO; SPI; I2C; I2S;
Розрядність АЦП [біт]	10
Розрядність ШІМ [біт]	10
Номінальна напруга [В]	3.3
Розміри [мм]	58*30

Digispark ATtiny85 – це дуже крихітна плата із невеликими можливостями, проте вона може стати в нагоді для створення маленьких дистанційно керованих систем, де перш за все потрібна компактність. Тут відсутні зайві виводи та роз'єми, лише все необхідне для роботи. Проте дана плата не може підключатися до комп'ютера напряму і потребує спеціального програматору, також тут відсутній захист на платі від неправильного підключення, але все це компенсується розмірами плати, до того ж на платі присутній стабілізатор напруги, що дозволяє підключати її напряму до батареї чи акумуляторів.

Приведемо основні характеристики (табл. 2.3):

Таблиця 2.3 –Характеристики Digispark ATtiny85

Компонент	**Характеристики**
Мікроконтролер	ATTINY85
Тактова частота [МГц]	8
Об'єм FLASH [кБ]	8
Об'єм SRAM [кБ]	0.5
Об'єм EEPROM [кБ]	0.5
Цифрові порти	6
Портів АЦП	4
Портів ШІМ	3
Максимальний струм на порт [мА]	40
Інтерфейси	USB; SPI; I2C
Розрядність АЦП [біт]	10
Розрядність ШІМ [біт]	8
Номінальна напруга [В]	5
Розміри [мм]	23*17.5

Arduino Nano Every – це сучасна плата, що стала розвитком Arduino nano, маючи той самий форм-фактор вона стала більш потужною та відповідає сучасним потребам. Одним із головних змін стало те, що тепер з’явився спеціальний блок (Core Independent Peripherals, CIP), що обслуговує периферію, цим сам звільняючи основне ядро від функцій постійного опитування та переривань. В усьому ж іншому вона майже повністю сумісна із платами попереднього покоління. Розробники плати також обіцяють більшу енергоефективність, що є одним із важливих факторів для автономної самохідної платформи.

Приведемо основні характеристики (табл. 2.4):

Таблиця 2.4 – Характеристики Arduino Nano Every

Компонент	**Характеристики**
Мікроконтролер	ATmega4809
Тактова частота [МГц]	16
Об’єм FLASH [кБ]	48
Об’єм SRAM [кБ]	6
Об’єм EEPROM [кБ]	0.25
Цифрові порти	22
Портів АЦП	8
Портів ШІМ	5
Максимальний струм на порт [мА]	20
Інтерфейси	USB; UART; SPI; I2C
Розрядність АЦП [біт]	10
Розрядність ШІМ [біт]	8
Номінальна напруга [В]	5
Розміри [мм]	45*18

Arduino Mega 2560 – ця досить велика плата є розвитком плати Arduino Uno, для тих кому не вистачає портів чи потужності, її досить часто використовують там де потрібно підключати багато датчиків чи контролювати складні процеси, це можливо оскільки тут встановлене досить потужне обчислювальне ядро. Сама по собі плата, сумісна із попереднім поколінням плат, живлення, напруги та принципи взаємодії аналогічні попереднім моделям.

Таку плату можна використовувати для самохідних платформ, де наявна велика кількість сенсорів або є потреба у потужності обрахунків.

Приведемо основні характеристики (табл. 2.5):

Таблиця 2.5 –Характеристики Arduino Mega 2560

Компонент	**Характеристики**
Мікроконтролер	ATmega2560
Тактова частота [МГц]	16
Об’єм FLASH [кБ]	256
Об’єм SRAM [кБ]	8
Об’єм EEPROM [кБ]	4
Цифрові порти	70
Портів АЦП	16
Портів ШІМ	15
Максимальний струм на порт [мА]	40
Інтерфейси	USB; 4 UART; I2C; SPI
Розрядність АЦП [біт]	10
Розрядність ШІМ [біт]	8
Номінальна напруга [В]	5
Розміри [мм]	102*53

Arduino Due – дана плата є подальшим розвитком Arduino Mega 2560, маючи вже 32-бітну архітектуру дозволяє швидко виконувати розрахунки, порівняно із попереднім поколінням. Потрібно мати до уваги те, що дана плата вже працює від 3.3В, а не від 5В як попередні плати. Із особливостей є те, що мікроконтролер має можливість прямого доступу до пам’яті, що може бути корисним для задач, котрі активно працюють із пам’яттю. Також плата має досить точний аналого-цифровий перетворювач. На платі є 2 USB виходи, один із яких може працювати в ролі хоста для під’єднаних пристроїв

Приведемо основні характеристики (табл. 2.6):

Таблиця 2.6 – Характеристики Arduino Due

Компонент	**Характеристики**
Мікроконтролер	AT91SAM3X8E
Тактова частота [МГц]	84
Об’єм FLASH [кБ]	512
Об’єм SRAM [кБ]	96
Об’єм EEPROM [кБ]	-
Цифрові порти	54
Портів АЦП	12
Портів ШІМ	12
Максимальний струм на порт [мА]	15
Інтерфейси	USB; 2 I2C; 4 UART; USB OTG; SPI; CAN
Розрядність АЦП [біт]	12
Розрядність ШІМ [біт]	12
Номінальна напруга [В]	3.3
Розміри [мм]	101*53

ESP32 WROOM DevKit – дана плата продовжує ідеї плати на основі ESP8266, вона може виконувати широкий спектр задач від потужної обробки мережевих сенсорів до обробки звуку чи кодування голосу. Перевагами даної плати є два обчислювальних ядра, одне із яких займається мережевими комунікаціями, а інше для виконання програми, що дозволяє отримати гарну якість зв'язку та мінімум затримок у основній програмі. Також розробник заявляє, що у даній платі наявні гнучкі схеми енергозбереження, але головними перевагами є те, що на платі наявні можливості для Wi-Fi та Bluetooth зв'язку, що при розробці самохідної, телеметричної платформи є дуже доречним.

Приведемо основні характеристики (табл. 2.7):

Таблиця 2.7 – Характеристики ESP32 WROOM DevKit

Компонент	**Характеристики**
Мікроконтролер	ESP32
Тактова частота [МГц]	240
Wi-Fi	2.4 ГГц 802.11 b/g/n/d/e/i/k/r
Bluetooth	Bluetooth v4.2 BR/EDR/BLE
Об'єм FLASH [кБ]	448
Об'єм SRAM [кБ]	520
Об'єм зовнішньої FLASH [кБ]	4096
Цифрові порти	21
Портів АЦП	15
Портів ЦАП	2
Портів ШІМ	21
Максимальний струм на порт [мА]	40
Інтерфейси	3 UART, 3 SPI, SDIO,3 I2S, 2 I2C, IR
Розрядність АЦП [біт]	12
Розрядність ШІМ [біт]	16
Номінальна напруга [В]	3,3
Розміри [мм]	51*28

STM32 Nucleo-64 F401RE – ця плата являє собою плату розробника відомого виробника мікроконтролерів STMicroelectronics і являє собою, більш орієнтовану на промисловість плату, ніж Arduino чи ESP. Хоча мікроконтролер є 3.3В проте порти плати підтримують і 5В логіку. Головною особливістю є те, що дане сімейство мікроконтролерів є одним із промисловим стандартів, що є перевагою і недоліком, так як накладає підвищені вимоги на розробника, оскільки потрібно більше розбиратися на низькому рівні ніж при роботі із іншими популярними платами. Із переваг гарна потужність, велика стабільність роботи та кількість підтримуваних технологій. Також дана плата є сумісною із модулями для Arduino.

Приведемо основні характеристики (табл. 2.8):

Таблиця 2.8 – Характеристики STM32 Nucleo-64 F401RE

Компонент	**Характеристики**
Мікроконтролер	STM32F401RET6
Ядро	Cortex-M4
Тактова частота [МГц]	84
Об’єм FLASH [кБ]	512
Об’єм SRAM [кБ]	96
Цифрові порти	81
Портів АЦП	10
Портів ШІМ	32
Максимальний струм на порт [мА]	25
Інтерфейси	4 SPI; 3 I2C; 3 USART; SDIO; USB 2.0 OTG
Розрядність АЦП [біт]	12
Розрядність ШІМ [біт]	16
Номінальна напруга [В]	3.3
Розміри [мм]	82.5*70

Для віддалено керованих платформ із великою кількістю сенсорів та задач які потребують багато обчислювальної потужності, наприклад нейронні мережі, чи просунутий технічний зір доцільно буде використання міні комп'ютерів в якості центрального керуючого блоку. Загалом одноплатний комп'ютер маю досить надлишкові можливості та потужності для проектування самохідних платформ, про те там де він справді потрібні він є незамінним.

Розглянемо популярне рішення від виробників для цього напрямку.

Raspberry Pi 4 Model B – це невеликий (розміром с кредитну карту) одноплатний комп'ютер. Може виконувати повноцінну операційну систему таку як Linux чи Windows. Із переваг потрібно наголосити на внутрішніх модулях Wi-Fi та Bluetooth, декілька роз'ємів USB у різних форм-факторах, роз'єм Ethernet, два HDMI роз'єми. В одній лінійці продукту, є декілька моделей на різну комплектацію, також наявна сумісність із попередніми поколіннями, що дозволяє вибирати плату виходячи із поставлених перед платформою задач.

У даної плати дуже розвинена спільнота та гарна технічна документація, що спрощує вирішення більшості проблем у розробці. Не варто також забувати, що плата маючи досить потужний процесор інколи потребує невеликого активного охолодження.

Приведемо коротко основні характеристики (табл. 2.9):

Таблиця 2.9 –Характеристики Raspberry Pi 4 Model B

Компонент	**Характеристики**
Процесор	4-х ядерний 1.5 ГГц BCM2711 Cortex-A72
Об'єм оперативної пам'яті [ГБ]	від 1 до 8
Об'єм постійної пам'яті [ГБ]	в залежності від flash карти
Комунікаційні можливості	Wi-Fi – 2.4 та 5 ГГц 802.11ac; Bluetooth 5.0; Гігабітний Ethernet
Інтерфейси	2*USB2.0; 2*USB3.0, I2C, I2S, SPI/UART
Портів вводу/виводу	40
Напруга живлення [В]	5
Розміри [мм]	86*56

2.2.2 Датчики

Датчики поділяються на аналогові та цифрові (дискретні).

Аналогові передають мікроконтролеру саму фізичну величину, яка неперервно характеризує досліджуваний об’єкт, зазвичай це або цілі числа, або дробові.

Цифрові або дискретні датчики, вимірюють величину, можуть її нормалізувати, приводять до шкали вимірювання, проводиться калібрування та інші потрібні обрахунки. Після цього величина кодується у двійковий код та передається будь-яким цифровим інтерфейсом до мікроконтролера.

Велике різноманіття датчиків не дозволяє їх усіх однозначно класифікувати, проте можна виділити найбільш популярні типи датчиків.

- Датчики світла – реагують на світло, можуть лише показати його наявність чи відсутність або дуже точно вимірювати його інтенсивність.
- Датчики звуку – дозволяють записувати, обраховувати та реагувати на звукові коливання, це можуть бути реакції на шум
- Датчики орієнтації – дають виміри лінійної та кутової швидкості, що дозволяє створити представлення про положення роботизованої системи.
- Датчики клімату – вид датчиків, що дозволяють отримувати значення температури, вологості, тиску, наявність окремих видів газу чи диму.
- Датчики простору – найпростіший вид машинного зору, дозволяє отримати дистанцію до об’єктів або хоча би факт їх присутності поряд.
- Датчики керування – це різноманітні кнопки, джойстики, потенціометри або сенсорні кнопки, тобто усе, що здатне перетворювати маніпуляції людини в електричний сигнал.
- Датчики механічного впливу – ці датчики здатні виявляти коливання фізичного впливу на об’єкти наприклад, поштовхи, удари і т.д.
- Датчики електромагнітних величин – вимірюють напругу, струм, напруженість магнітного поля і т.д. Найпростіші датчики, які лежать в основі усіх інших.

Згідно із вимогами до СТРП (підрозділ 2.1) розглянемо асортимент можливого обладнання, які представлені на сучасному ринку.

Для вимірювання вологості та температури, зазвичай використовують спільні модулі, що можуть вимірювати ці обидва показника, на модулі вже встановлені самі датчики, та конвертер, що перетворює аналоговий сигнал у цифровий. У таких модулях є вже калібрування із заводу. Розглянемо їх характеристики у таблиці нижче (табл. 2.10).

Таблиця 2.10 – Характеристики датчиків вологості та температури

Датчик	Діапазон вологості, %	Діапазон температури, °C	Час між опитуванням, с	Точність вологості, %	Точність температури, °C
DHT-11	20-90	0-50	1	± 5	± 1
AM2320	1-99	-40 +80	2	± 3	± 0.1
SHT10	0-100	-40 +128	1	± 2	± 0.3
GY-21 HTU21	0-100	-40 +125	1	± 3	± 0.4
SHT30	0-100	0-65	8	± 3	± 0.3

Для вимірювання дальності до перешкоди використовують різноманітні далекоміри, це можуть бути лазерні, світлові чи ультразвукові далекоміри.

Розглянемо їх характеристики у таблиці нижче (табл. 2.11).

Таблиця 2.11 – Характеристики далекомірів

Датчик	Середовище	Діапазон вимірювань, мм	Точність	Кут виявлення перешкоди, °
HC-SR04/ US-025	Ультразвук	20-4000	3 мм	30
TOF10120	Лазер 940 нм	100-1800	5%	-
GP2Y0A41SK0F	Інфрачервоне світло	40-300	10%	40
VL53L0X-V2	Лазер 940 нм	50-2000	7%	-
YL-63	Інфрачервоне світло	20-300	10%	35
VL6180X	Лазер 850 нм	1-100	1%	-
HRLV-MaxSonar	Ультразвук	300-5000	1 мм	44

Для вимірювання швидкості у таких системах, як СТРП зазвичай використовують різноманітні оптичні чи магнітні енкодери, що ставляться на одне або декілька коліс. Є декілька способів це зробити:

- Перший спосіб це – пара фотоприймача та світлодіоду, між ним колесо із щілинами, кількість прийнятих сигналів и радіус колеса дають змогу дізнатися про швидкість.
- Другий спосіб – аналогічний першому тільки використовуються магніт та датчик (наприклад датчик Холла), що реагує на магнітне поле.

Беручи до уваги те, що такі датчики зазвичай збираються із різних дискретних компонентів та індивідуально для кожного випадку, відпадає потреба у їх детальному розгляді.

Для визначення атмосферного тиску використовуються цифрові барометри. Розглянемо їх характеристики у таблиці нижче (табл. 2.12).

Таблиця 2.12 – Характеристики цифрових барометрів

Датчик	Діапазон виміру тиску, гПа	Точність, Па
BMP-180	300-1100	1
BMP-280	300-1100	0.16
HX-710B	0-400	1
MS-5611	300-1100	1
GY-68	300-1100	3

Для визначення орієнтації роботизованої системи у просторі використовують, так датчикі як: акселерометри, що визначає прискорення тіла та гіроскопи, що визначає кут нахилу. Зазвичай вони працюють разом у вигляді єдиного модулю.

Розглянемо їх характеристики у таблиці (табл. 2.13).

Таблиця 2.13 – Характеристики акселерометрів та гіроскопів

Датчик	Діапазон гіроскопу, °/с	Діапазон акселерометру, g	Магнітометр
BNO055	± 125 до ± 2000	± 2; ± 4; ± 8; ± 16;	+
MPU-6050	±250; ±500; ±1000 ±2000;	± 2; ± 4; ± 8; ± 16;	-
LSM303C	-	± 2; ± 4; ± 8;	+
ADXL345	-	±16	-

2.2.3 Двигуни та їх драйвери

Для пересування самохідних дистанційно керованих систем, типовим рішенням є використання електричного, колекторного мікродвигуна із редуктором, який перетворює швидкі оберти у більш низькі, створюючи при цьому більший обертальний момент на валу приводу.

Драйвер для колекторного двигуна представляє собою Н-міст, із транзисторів. Вони дозволяють керувати напрямком обертання двигуна, змінюючи полярність струму, якщо ж подати ШІМ сигнал на драйвер, то можна керувати швидкістю обертання двигуна. Драйвери у першу чергу характеризуються, своєю робочою напругою та максимальним струмом, що визначає максимальну потужність, яку вони можуть надати для двигуна.

Розглянемо характеристики драйверів у таблиці нижче (табл. 2.14).

Таблиця 2.14 – Характеристики драйверів для колекторних мікродвигунів

Драйвер	Мінімальна напруга живлення, В	Максимальна напруга живлення, В	Робочий струм, А	Піковий струм, А	Кількість каналів
DRV8833	2.7	10.8	0.5	2	2
TB6612FNG	2.5	15	1.2	3.2	2
L298N	5	35	2	3	2
DRV8838	0	11	1.7	1.8	1
L9110S	2.5	12	0.8	1.2	2
VNH2SP30	5.5	16	14	30	1

2.2.4 Елементи живлення

У випадку поставлених перед СТРП вимог (підрозділ 2.1), доцільно розглядати лише елементи живлення, що забезпечують автономність. До таких елементів відносяться батареї та акумулятори. З огляду на те, що використання одноразової батареї не можливо у зв'язку із їх відсутністю достатньо потужних, дешевих та ємких для потреб віддалено керованої системи. Виходячи із цього будемо розглядати лише акумулятори.

Для початку розглянемо типи акумуляторів.

Свинцево-кислотні акумулятори (Pb) – це найбільш розповсюджений та простий тип акумулятору. Із переваг мають великі дуже високі токи розряду, особливо короткочасні, одні із самих дешевих, простий процес зарядки. Недоліком є невелика питома ємність.

Літій-полімерні акумулятори (Li-Po) – тип акумуляторів, характеризуються можливістю створення акумуляторів майже будь-якої форми, більш енергоефективні, ніж свинцево-кислотні, проте і дорожче. Із недоліків, мають складну зарядку та знижують свої характеристики на холоді.

Літій-іонні акумулятори (Li-ion) – один із найрозповсюджених типів акумуляторів. Мають високу питому ємність, низький саморозряд, середні токи розряду, не великий саморозряд.

Літій-залізо-фосфатні акумулятори (LiFePo4) – відносно нові акумулятори, що мають ряд переваг, такі як: швидкий заряд, дуже великі струми розряду, можливість роботи на холоді, велика кількість робочих циклів. Недоліки: одні із найважчих із літієвих акумуляторів, потребують спеціального зарядного пристрою.

Нікель-кадмієві акумулятори (Ni-Cd) – один із найстаріших типів акумуляторів, характеризуються високими токами розряду, відсутністю шкідливих ефектів при надмірному розряді чи заряді, проста та швидка зарядка, можуть зберігатися розрядженими. Головний недолік мають ефект пам'яті, коли їх заряджають раніше, ніж вони повністю розрядились .

Нікель-метал-гідридний акумулятор (Ni-MH) – покращена версія нікель кадмієвих акумуляторів, вони не мають ефекту пам’яті, мають просту зарядку, немає шкідливих ефектів при надмірному заряджанні та розряджанні, проте головний недолік те що вони не підтримують великі струми розряду.

Розглянемо розповсюджені моделі у таблиці нижче (табл. 2.15).

Таблиця 2.15 – Характеристики моделей акумуляторів

Модель	Тип	Ємність, мА*год	Макс. струм розряду, мА	Струм розряду, мА	Номінальна напруга, В	Формфактор
NCR18650B	Li-ion	3200	4800	1625	3.6	18650
Li-Po303040	Li-Po	350	300	150	3.7	303040
LG18650HG2	Li-ion	3000	20000	1500	3.6	18650
INR18650P	Li-ion	2000	20000	2000	3.7	18650
HR-3UTC	NiMH	1900	900	570	1.2	AA
Li-Po103040	Li-Po	1200	1000	500	3.7	103040

2.2.5 Телекомунікаційне обладнання

Для дистанційно керованої системи, одним із найважливішим компонентів є такий, що надає можливості бездротового зв’язку. Зв’язок цей може бути організований по радіоканалу різними технологіями такими як:

- Wi-Fi – надає можливості високошвидкісного зв’язку та низької затримки, має можливості для підключення великої кількості приладів, має шифрування інформації, працює на 2.4 та 5 ГГц частотах. Проте для автономних систем значним недоліком є досить високе споживання енергії.
- Bluetooth – низькошвидкісний зв’язок, підтримує множинне підключення, надає захист передачі інформації, створений для не великої дальності – до 100 м (на практиці 10-20 м), і будь-яка перешкода значно знижує дальність. Головною перевагою є низьке споживання інформації.

- GSM – метод зв’язку, що дозволяє користуватися стільниковим зв’язком і має ті ж самі переваги і недоліки, що і GSM мережі. Має високе споживання енергії.
- ZigBee – протокол бездротового зв’язку, що має високий захист від завад, дуже низьке енергоспоживання, високу швидкість пробудження із «режиму сну», може створювати не тільки базові топології мережі (дерево, зірка, точка-точка), а і само організовану коміркову (mesh) топологію, із ретрансляцією повідомлень і разі виходу одного із вузлів, можуть відновити працездатність мережі.
- LoRa – технологія низькошвидкісної мережі із принципом передачі даних, що оснований на розширені спектру та лінійній частотній модуляції дозволяє демодулювати дуже слабкий сигнал (-148 дБ), це дозволяє вести передачу даних на великі відстані із низьким споживанням енергії.
- Радіоканал – це передача даних без протоколів, або по спрощеним протоколам, на частотах, що не ліцензуються (443 МГц, 2.4 ГГц і т.д), мають недоліки та переваги в залежності від свого виконання.

Потрібно також зауважити, що деякі модулі зв’язку, мають у своєму складі також потужний мікропроцесор та порти вводу/виводу, що дозволяє використовувати їх не лише для організації зв’язку, а і у якості мікроконтролеру загального призначення з деякими обмеженнями.

Розглянемо сучасні популярні рішення для організації бездротового зв’язку на віддалено керованих платформах аналогічним СТПР.

TTGO T-Call ESP32 SIM800L – універсальна плата, її головною перевагою є те, що вона містить у собі Wi-Fi, Bluetooth та GSM модуль, які надають, аж три різні шляхи для створення бездротового зв’язку.

Приведемо коротко основні характеристики:

- Wi-Fi – 802.11 b/g/n, потужність передавача 22 дБм, макс. дальність зв’язку 300 м, швидкість передачі 300 Мбіт/с.
- Bluetooth – v4.2BR/EDR/BLE, чутливість приймача -97 дБм.

- GSM – GSM/GPRS діапазони: GSM 850 МГц, EGSM 900 МГц, DCS 1800 МГц, PCS 1900 МГц. Швидкість передачі 85.6 Кбіт/с.

ESP32 – компактний модуль бездротового зв’язку від відомого виробника Espressif, який підтримує Wi-Fi та Bluetooth. Маючи гарну вбудовану антену, є можливість під’єднання зовнішньої.

Приведемо коротко основні характеристики:

- Wi-Fi – 2.4ГГц 802.11 - b/g/n/d/e/i/k/r .
- Bluetooth - v4.2BR/EDR/BLE, чутливість приймача -98 дБм.

Bluetooth CC254 – модуль, що реалізує можливості Bluetooth Low Energy (BLE), характеризується невеликим споживанням енергії у режимі передачі.

Приведемо коротко основні характеристики:

- Частота: 2.4 ГГц, швидкість передачі 1Мбіт, дальність зв’язку до 60 м.
- Потужність передавача: 0 дБм, чутливість приймача -94 дБм.

GSM SIM800C – модуль стільникової мережі надає повноцінні 4 діапазони GSM/GPRS. Може обмінюватися текстовими та голосовими повідомленнями через мережу. Також на платі присутній Bluetooth версії 3.0.

Приведемо коротко основні характеристики:

- Частотні діапазони: 850;900;1800;1900 МГц.
- Максимальна швидкість: 85.6 Кбіт/с.

Sparkfun XBee 3 – модуль, надає можливості стандарту 802.15.4 та протоколу ZigBee, окрім того є можливість користуватися протоколом Bluetooth BLE. Ці модулі спеціально створені, що бути максимально енергоефективними, для низькошвидкісних мереж.

Приведемо коротко основні характеристики:

- Швидкість передачі даних: 250 Кбіт/с, дальність передачі: до 1200 м.
- Потужність передавача: 8 дБм, чутливість приймача: -103 дБм.
- Шифрування 128/256 біт AES.

LoRa Module Ra-02 – модуль зв'язку, із невеликою потужністю, що реалізую протокол Long Range (LoRa). Головною особливістю є велика дальність зв’язку при невеликій потужності.

Приведемо коротко основні характеристики:

- Потужність передавача 20 дБм, чутливість приймача -148 дБм.
- Швидкість передачі: 300 Кбіт/с, частотний діапазон: 443 МГц.

NRF24L01P+ – модуль, що працює на частоті 2.4 ГГц. За допомогою зовнішньої антени та підсилювача може надавати дальність передачі до 1100 м. Підтримує топологію мережі типу «зірка».

Приведемо коротко основні характеристики:

- Потужність передавача 20 дБм, чутливість приймача -95 дБм.
- Швидкість передачі: 2 Мбіт/с, частотний діапазон: 2.4 ГГц.

2.2.6 Відео обладнання

Із опису СТРП (підрозділ 2.1) можна побачити, що є потреба у передачі потокового відео. Такі можливості надають спеціальні модулі, на яких встановлені лінзи та камера. Зазвичай такі камери оснащені ширококутною лінзою та спеціальним мікроконтролером, який кодує та перенаправляє готові дані до мікроконтролеру через інтерфейс.

Розглянемо декілька популярних рішень для потокової передачі відео у таблиці нижче (табл. 2.16)

Таблиця 2.16 – Характеристики FPV камер

Модуль камери	Максимальна роздільна здатність, піксель	Кут лінзи	Інтерфейс	Відношення сигнал/шум, дБ
OV2640	1600*1200	120	SCCB	40
PTC06	640*480	120	TTL	45
1200TVL	1960*1280	150	CVS	52
OV7670	640*480	25	SCCB	46

2.3 Обладнання та компоненти дистанційно керованої платформи

Даний підрозділ включає в себе обґрунтування вибору того чи іншого обладнання, згідно опису завдання (підрозділ 2.1) та проведеного аналізу асортименту можливого обладнання із підрозділу 2.2.

Буде обрано конкретні рішення, для виконання функцій СТРП: блоку керування, приймально-передавального пристрою, датчиків, блоку живлення, шасі та корпусу. У ході обґрунтування цих рішень будуть описані детально їхні характеристики.

Надалі у тексті зустрічаються посилання словами «було розглянуто», «проаналізовано» і тому подібне, що говорить про інформацію із підрозділу 2.2 та посилання словами «опис завдання», «вимоги» і т. п. , відповідно із розділу 2.1.

2.3.1 Блок керування

На роль блоку керування було розглянуто 10 видів обладнання 2 із яких є міні-комп’ютерами.

Міні-комп’ютери Raspberry Pi та Orange Pi – детально не розглядалися, оскільки вони мають надмірну потужність для побудови такої платформи, як СТРП, із за чого мають такі недоліки як:

- Не ефективне використання обладнання – обчислювальні можливості можуть простоювати більшу частину часу.
- Наявність великої кількості інтерфейсів, потреби у котрих немає.
- Досить висока вартість – порівняно із більш простими рішеннями;
- Більше споживання енергії, що зменшує час автономної роботи.

Проте, якщо була би потреба у високоякісній трансляції відео, ці міні-комп’ютери були б ідеальним рішенням, оскільки кодування відео та його трансляція потребує значних обчислювальних можливостей. Також у такому обладнанні могла виникнути потреба під час створення платформи із просунутим штучним інтелектом, тобто такі рішення доцільно використовувати коли потрібні складні обрахунки, а на СТРП такої потреби немає.

Далі було розглянуто такі плати розробників як: NodeMCU ESP8266 та ESP32 WROOM DevKit.

Ці плати хоч і мають універсальний мікроконтролер, все ж їх призначення це організація зв'язку. І в якості блоку керування їх використання не доцільне оскільки в них не достатня кількість портів вводу/виводу та портів, що мають функції АЦП, що є дуже важливим для такої платформи як СТРП, оскільки саме порти та їх функції надають можливість підключення більшої кількості датчиків.

До того ж варіант плати із мікроконтролером ESP32 буде використовуватися у складі СТРП, як блок, що забезпечує комунікацію. Враховуючи все вище сказане, ці дві плати не доцільно використовувати в ролі блоку керування.

Потім було розглянуто плату на базі STM32 – STM32 Nucleo-64 F401RE, хоча ця плата є більш «дорослою» у світі мікроконтролерів та має кращі показники якість/ціна, стабільності у роботі, проте принцип роботи із нею відрізняється від інших плат і потребує більше зусиль розробника для створення програми та її супроводу. Беручи до уваги, що СТРП є прототипом, не доцільно використовувати таку складну для розробника платформу, адже для створення прототипів одним із головних є швидкість побудови, коли можна по різному компонувати обладнання та перевіряти його на практиці.

Плати Arduino Mega 2560 та Arduino Due, добре підходять для розробки телеметричних платформ, зважаючи на їх збільшену порівняно із молодшими моделями обчислювальну потужність та наявність великої кількості портів, як звичайних так і АЦП та ШІМ. Проте для СТРП відсутня потреба у таких платах оскільки немає такої кількості датчиків та виконавчих пристроїв. До того ж вони мають більші габаритні розміри. Тому використовувати дані плати не доцільно.

У ході аналізу обладнання також було розглянуто плату Digispark ATtiny85, вона має дуже компактний форм-фактор, що дуже доречно для маленьких моніторингових систем, проте головним недоліком є мала кількість портів, що не відповідає завданню СТРП. Отже Digispark ATtiny85, не є доцільно використовувати.

На розгляд також потрапила плата Arduino Nano Every покращена версія Arduino Nano, маючи більш потужне ядро, більше пам’яті та портів вводу/виводу. Проте в неї мале відношення ціна/можливості, адже її ціна у 4-5 разів вища за попереднє покоління. До того ж плата відносно нова, як мікроконтролер у її складі, тому деякі речі можуть працювати із помилками на ній, потребуючи доопрацювання. Використання такої плати є не доцільним.

Узагальнюючи усе вище сказане, із можливого асортименту обладнання для створення СТПР найкраще підходить плата розробника Arduino Nano або Arduino Uno. Оскільки обидві ці плати є аналогічними за своїми можливостями і відрізняються між собою лише формфактором, перевагу у виборі буде надано Arduino Nano, оскільки вона менша за розмірами, а отже добре підходить для мобільної платформи.

Arduino Nano – має довгу історію на ринку (з 2008 р), отже добре протестована спільнотою, що надає перевагу у швидкій розробці прототипу СТРП.

У цієї плати 14 цифрових портів вводу/виводу, 8 із яких можуть бути під’єднані до АЦП, що дозволяє працювати із усіма датчиками, що будуть встановлені, а також залишиться запас на можливу модернізацію платформи.

Сила струму на виходах може досягати 40 мА, що дозволяє підключати напряму транзистори або деяке обладнання. На платі присутній стабілізатор напруги що дозволяє надавати живлення у діапазоні від 7 до 12 В.

Для виконання програми мікроконтролер має:

- 16 МГц тактової частоти.
- 32 кБ пам’яті для програмного коду.
- 2 кБ оперативної пам’яті.
- 1 кБ енергонезалежної пам’яті.

Даних характеристик цілком достатньо для виконання усіх поставлених перед СТРП завдань.

Рисунок 2.3 – Плата розробника Arduino nano

2.3.2 Датчики

Для вимірювання вологості та температури був обраний гібридний датчик DHT-11, оскільки СТРП не має перед собою реальні практичні цілі, а більше як демонстраційний зразок такого датчика цілком достатньо.

Приведемо його характеристики:

- Напруга живлення: 3 - 5 В.
- Діапазон вологості: 20-90 ±5%.
- Діапазон температури: 0-50 ±1 °C.
- Час між опитуваннями: 1 с.

DHT-11 у вигляді модуля має три виводи 1 – для живлення, 2 – для даних, 3 – для землі (GND).

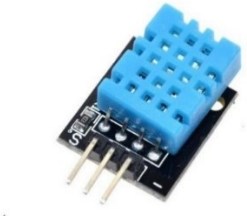


Рисунок 2.4 – Датчик вологості та температури DHT-11

У якості датчику, що виявляє перешкоду було розглянуто 7 варіантів різних далекомірів, окрім точності та дальності вони відрізнялися і фізичним принципом виявлення перешкоди, а саме середовищем. Для СТПР було обрано саме ультразвук, оскільки він має широкий кут направленості, що дозволяє встановити лише один датчик для отримання досить точної інформації про наявність об’єктів перед самохідною платформою.

На роль далекоміра було обрано модуль датчику US-025, це повний аналог датчику HC-SR04, що має велику популярність серед спільноти розробників.

Точності та дальності US-025 достатньо для виконання СТРП поставлених задач.

Приведемо його характеристики:

- Напруга живлення 3 - 5.5 В.
- Максимальний та ефективний кут огляду: 30/15°.
- Діапазон вимірювань 20-4000 мм.

US-025 має 4 виводи: 1 – керуючий, 2 – для збирання інформації, і два для живлення та землі (GND).

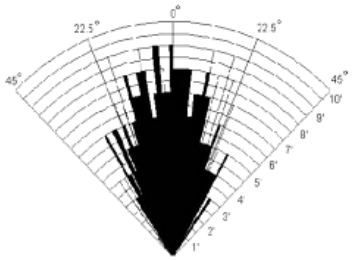


Рисунок 2.5 – Діаграма направленості ультразвукового датчику US-025

Рисунок 2.6 – Ультразвуковий далекомір US-025

Для вимірювання атмосферного тиску було обрано датчик BMP-280, оскільки він має гарні характеристики та більш доступний на ринку, ніж інші датчики. Має внутрішній термометр для компенсації впливу температури на вимірювання.

Приведемо його характеристики:

- Напруга живлення: 1.7 – 3.6 В.
- Підтримувані інтерфейси: I2C, SPI.
- Діапазон вимірювань: 300 – 1100 гПа ±1 Па.

Крім того має можливість змінювати час між вимірюваннями, підтримує фільтрацію вимірювань та має три режими роботи для більш ефективного енергоспоживання.

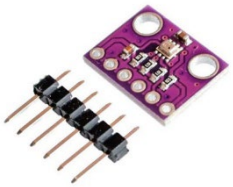


Рисунок 2.7 – Цифровий барометр BMP-280

Для орієнтації самохідної платформи у просторі необхідно використовувати акселерометр та гіроскоп, ці два датчика працюючи разом дозволяють отримувати значення кута нахилу. В якості такого датчику було обрано MPU-6050, він має і гіроскоп і акселерометр із гарними показниками точності та діапазонів вимірювань. До того ж має популярність у розробників, із чого є можливість користуватися бібліотеками програмного коду для роботи із датчиками.

Приведемо його характеристики:

- Напруга живлення: 3 - 5 В.
- Гіроскоп діапазон: ±250; ±500; ±1000 ±2000 °/с.
- Акселерометр діапазон : ± 2; ± 4; ± 8; ± 16 g.
- Підтримуваний інтерфейс: I2C.

MPU-6050 також має датчик температури для компенсації її впливу на вимірювання. Має власний контролер шини I2C, що надає можливість працювати у головному режимі.

Рисунок 2.8 – Модуль датчиків акселерометру та гіроскопу MPU-6050

Для вимірювання швидкості самохідної платформи буде використовуватися цифровий датчик Холла A3144, який буде встановлений біля ведучого колеса, що містить у собі пару неодимових магнітів. Швидкість обертання магнітів над датчиком Холла, дасть змогу визначити швидкість усієї платформи в цілому.

Для вимірювання напруги акумулятора на самохідній платформі, буде використовуватися, один із портів Arduino Nano, із функцією АЦП, що дозволить визначати залишок заряду у акумуляторах.

Для визначення сигналу від СТРП до терміналу буде використовуватися одна із вбудованих функцій Wi-Fi контролеру.

2.3.3 Комунікаційне обладнання

У якості модуля, що надає організацію бездротового зв’язку буде використовуватися мікроконтролер ESP-32, оскільки має потужні обчислювальні можливості, високошвидкісний зв’язок шляхом протоколу Wi-Fi, та має можливість Bluetooth зв'язку, що може знадобитися у якості резервного чи допоміжного каналу.

Приведемо його характеристики:

- Wi-Fi: 802.11 b/g/n, Bluetooth v4.2.
- Потужність передавача Wi-Fi: 19.5/16.5/15.5 дБм @11b/11g/11n.
- Чутливість приймача Wi-Fi: -98/-93/-75 дБм @1/6/11/54 Мбіт/с.

Рисунок 2.9 – Мікроконтролер ESP-32

2.3.4 Система живлення

Для забезпечення живлення напругою було обрано рішення, що включає в себе два літієвих акумулятори Westinghouse INR18650P форм-фактору 18650 ємністю 2000 мА*г, вони підключені паралельно і можуть надати до 20А струму при 3.7 В одночасно. Цього цілком досить для забезпечення потреб живлення СТРП.

Вибором саме літієвих акумуляторів стало, те що вони мають гарне відношення ємності до власної ваги, що є дуже важливим для автономності самохідних дистанційно керованих платформ.

Для того, що б можна було заряджати акумулятори, на платформі був встановлений модуль для заряджання літієвих акумуляторів із роз’ємом USB Type-C. Це надає змогу зручно заряджати акумулятори від звичайних побутових зарядок.

2.3.4 Шасі та корпус

На ринку наявне не дуже велике різноманіття варіантів гусеничного шасі. Тому було обрано варіант створити його на базі іграшкового танку, оскільки він уже має у своєму складі і шасі, і двигуни, і редуктори до них, що є досить зручним варіантом. Загальну будову шасі можна побачити на малюнку нижче (рис 2.8).

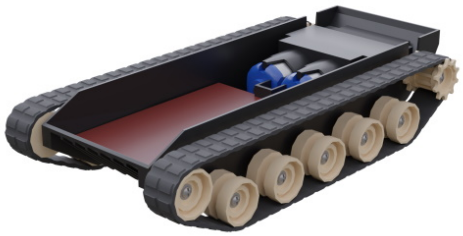


Рисунок 2.10 – Шасі для платформи СТРП

2.3.5 Відео обладнання

Для забезпечення платформи СТРП потоковим відео потрібна відеокамера досить гарної якості, але не надлишкової оскільки, більша якість відео потребує більшої обчислювальної потужності, витрат енергії та мережевого трафіку для передачі даних.

Варіантом, що виконує ці вимоги став модуль камери OV2640. Із головних переваг є те, що вона може підключися до мікроконтролеру ESP-32, що дозволяє по максимуму задіяти наявне обладнання.

Приведемо його характеристики:

- Напруга живлення 3В.
- Роздільна здатність: 1600*1200@15fps/800*600@30fps/352*288@60fps.
- Відношення сигнал/шум 40 дБ.
- Динамічний діапазон 50 дБ.

Рисунок 2.11 – Модуль камери OV2640

ВИСНОВОК

У другому розділі для визначення вимог до обладнання, було сформовані завдання, які повинна виконувати СТРП, завдяки цьому можливо було однозначно визначити необхідний набір компонентів із яких складається дистанційно керована платформа.

При виборі компонентів СТРП обрано плату розробника із мікроконтролером у якості блока керування та набір датчиків для виконання поставлених перед платформою завдань, а саме збирання потрібної телеметрії. Також був проведений огляд елементів живлення, відео та комунікаційного обладнання.

Кожний вибір того чи іншого обладнання обґрунтований виходячи із їх характеристик та потреб платформи. Пріоритет у проведенні аналізу сучасних компонентів віддавався мікроконтролерам, датчикам та комунікаційному обладнанню, оскільки саме ці компоненти характеризують платформу.

Обравши усі необхідні для побудови СТРП компоненти, можливо приступати до наступного розділу якому буде описаний процес створення самохідної телеметричної платформи.

3. СТВОРЕННЯ ПЛАТФОРМИ, ДОДАТКУ КЕРУВАННЯ ТА РОЗРАХУНКИ ЇХ ХАРАКТЕРИСТИК

3.1 Створення телеметричної самохідної платформи

У даному розділі буде детально розібрано процес побудови самохідної платформи. Зокрема монтаж та електричне з’єднання компонентів.

3.1.1 Підготовка шасі

У якості шасі було використано нижню частину від іграшкового танка, разом із гусеницями та двигунами і редукторами. Для того, що б почати повноцінно користуватися даними шасі як базою для подальшої побудови СТРП, потрібно її підготувати, для цього було проведено наступні дії:

- Вирізані зайві пластикові перегородки та кріплення що заважали подальшому монтажу компонентів.
- Змащені усі деталі, що мають тертя, зокрема редуктор.
- Зроблена металева верхня відкидна кришка на петлі, для зручного доступу до внутрішньої частини самохідної платформи.
- Поставлені кронштейни та пластикові кріплення для друкованих плат на верхній кришці.

Загалом було проведено ряд робіт для подальшого монтажу компонентів.

3.1.2 Монтаж датчиків, мікроконтролерів та інших плат

Для зручного монтажу датчиків було обрану спеціальну друковану макетну плату розмірами 50*70 мм, вона має невеликі отвори через 2.54 мм кожний, що дозволяє вставляти та запаювати більшість електронних компонентів.

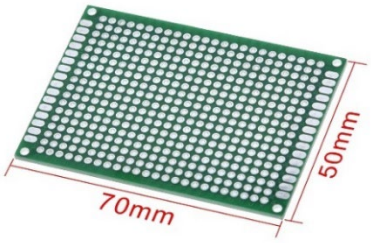


Рисунок 3.1 – Макетна плата 50*70 мм

На даній платі будуть розміщені: усі датчики, перемикач та схеми живлення.

Також на ній через кріплення буде встановлений спеціальний адаптер для плати розробника Arduino Nano. Він дозволяє підключити усі необхідні проводи до плати та водночас знімати її із плати для програмування та налаштувань.

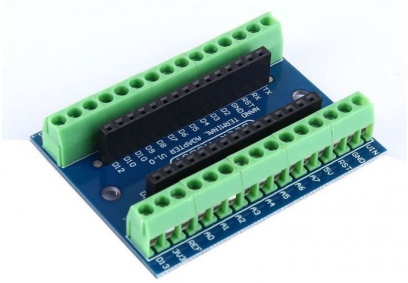


Рисунок 3.2 – Адаптер для Arduino Nano

Для ультразвукового далекоміру, що б правильного його зорієнтувати у просторі потрібно кріплення на передньому краю корпусу. Для цього був обраний акриловий кронштейн, він дозволяє надійно закріпити датчик на верхній кришці самохідної платформи.

Рисунок 3.3 – Кронштейн для ультразвукового датчику

3.1.3 Підключення датчиків

Датчик вологості та температури DHT-11 має 4 виводи із яких один не використовується, 2 виводи один із яких підключається до лінії живлення 5В інший до загального проводу (GND). Вивід «out» підключається до Arduino nano, по ньому передаються дані.

Електрична схема підключення датчику DHT-11 має наступний вигляд:

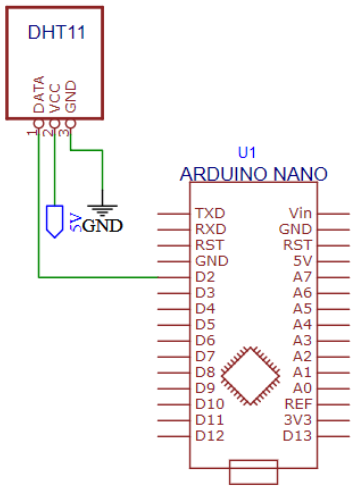


Рисунок 3.4 – Схема підключення DHT-11

Ультразвуковий датчик US-025 має 4 виводи, два для живлення і GND та два інших: «trigger» – для подання сигналу, і «echo» для його прийняття, вони підключаються до Arduino nano.

Електрична схема підключення датчику US-025 має наступний вигляд:

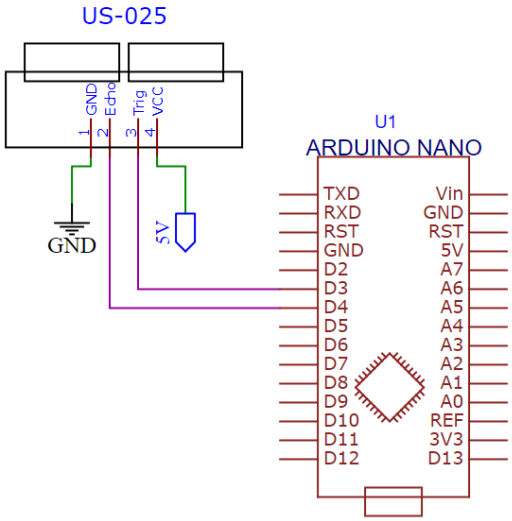


Рисунок 3.5 – Схема підключення US-025

Датчик Холла A3144 має два виводи для живлення і GND та один сигнальний. Для роботи датчика потрібно встановити резистор на 10 кОм між його сигнальним виводом та 5В, оскільки при спрацюванні на магніт даний датчик з’єднує сигнальну лінію із загальним проводом. І для забезпечення стабільного стану коли немає магніту повинен бути резистор до лінії живлення.

Електрична схема підключення датчику Холла має наступний вигляд:

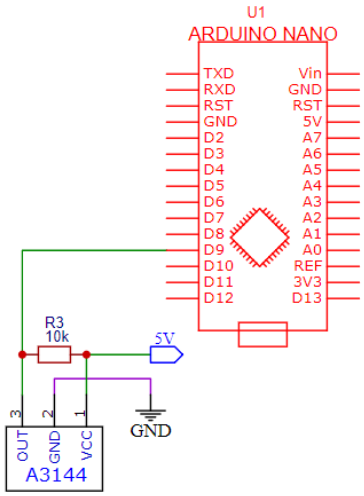


Рисунок 3.6 – Схема підключення A3144

Цифровий барометр BMP280 має 6 виводів із них два для живлення і GND, та 4 для інтерфейсів I2C та SPI. Для СТРП буде використовуватися інтерфейс I2C, оскільки він потребує меншої кількості провідників. Для інтерфейсу I2C використовуються виводи SCL та SDA, отже їх теж потрібно підключити до відповідних виводів на Arduino nano.

Електрична схема підключення барометру має наступний вигляд:

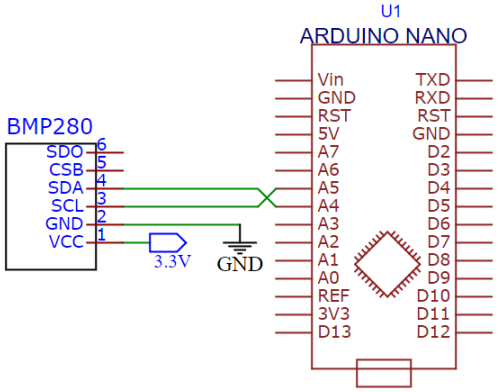


Рисунок 3.7 – Схема підключення BMP280

Модуль акселерометру та гіроскопу MPU-6050 має 8 виводів, два із яких призначенні для живлення та GND. Чотири виводи призначені для реалізації цифрових інтерфейсів I2C та SPI. Ще два виводи використовуються для зміщення адреси та переривання. Оскільки використовується інтерфейс I2C то будуть підключені виводи SCL та SDA до Arduino nano.

Електрична схема підключення акселерометру та гіроскопу має наступний вигляд:

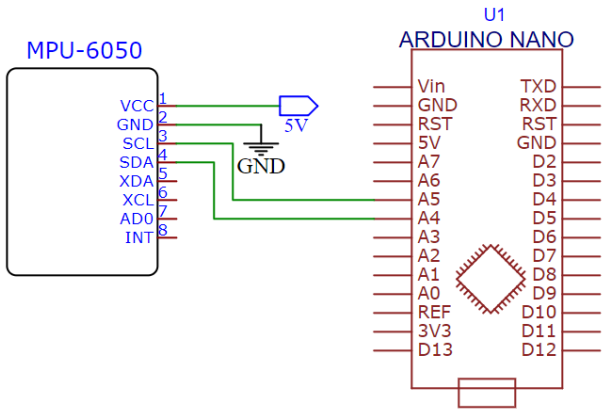


Рисунок 3.8 – Схема підключення MPU-6050

3.1.4 Підключення драйверу двигунів

Драйвер для колекторних двигунів L298N має 13 виводів:

- VSS, GND, Vs – для живлення двигунів, загальний провід та вхід для живлення логічної схеми (на платі є перемичка, завдяки якій можливо подавати живлення лише на вхід для двигунів, а внутрішній стабілізатор перетворить вхідну напругу до 5в для живлення логіки).
- IN1, IN2, IN3, IN4 – контакти для керування двома колекторними двигунами, або одним кроковим двигуном.
- ENA, ENB – контакти для подання ШІМ сигналу, що б керувати швидкістю двигунами, якщо ШІМ керування не потрібне, є перемичка для замикання виводу на 5В.

Електрична схема підключення драйверу для двигунів має наступний вигляд:

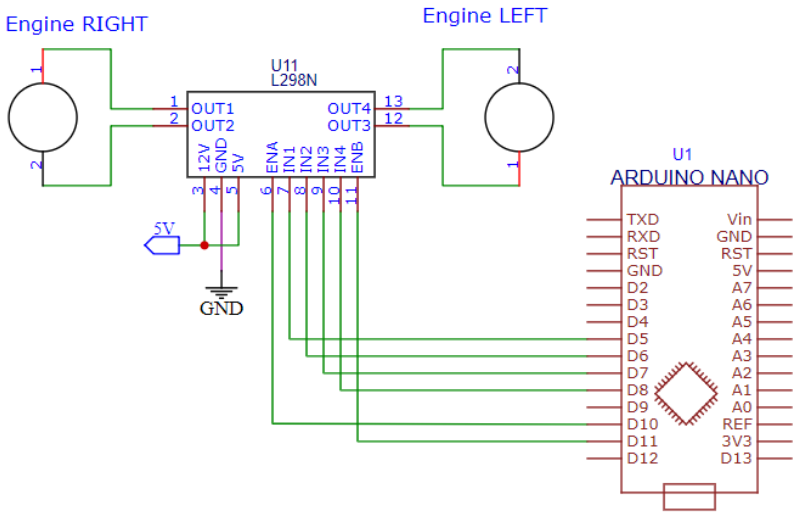


Рисунок 3.9 – Схема підключення L298N

3.1.5 Підключення лінії зв’язку між контролерами

Мікроконтролер ESP-32 виконує роль комунікаційного обладнання, забезпечуючи зв’язок із додатком керування, по протоколу Wi-Fi. А плата Arduino nano є керуючим блоком, який здійснює зняття показань із датчиків та управляє швидкістю і напрямком обертання двигунів. Отже потрібно передавати дані із датчиків на ESP-32 від Arduino nano і навпаки передавати від EPS-32 до Arduino nano сигнали керування. Тобто потрібно організувати двосторонній зв’язок між цими двома платами.

Зважаючи на невелику дальність передачі та можливість організації дротового з’єднання, найкращим рішенням для цього є протокол UART, він підтримується апаратно на обох платах та потребує лише двох дротів для з’єднання. Потрібно передавач однієї плати з’єднати із приймачем другої і так само навпаки.

ESP-32 має 3.3В логіку, а Arduino 5В, то необхідно використовувати конвертер логічних рівнів.

Електричну схему з’єднання мікроконтролерів можна побачити на малюнку нижче:

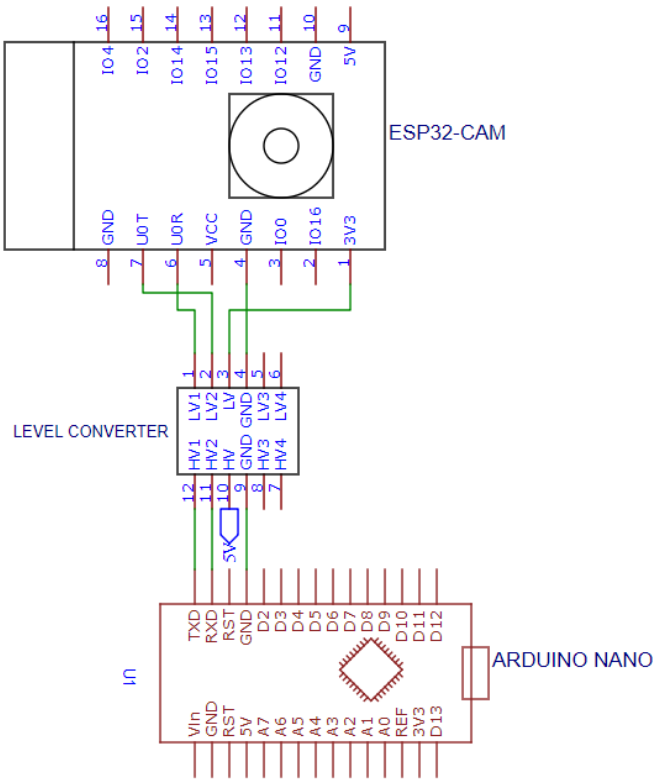


Рисунок 3.10 – Схема підключення інтерфейсу UART

3.2 Розрахунки параметрів платформи: автономності, дальності покриття та даних із сенсорів

3.2.1 Розрахунок дальності Wi-Fi зв'язку

Одним із найважливіших параметрів телеметричної самохідної дистанційно керованої платформи є дальність радіозв’язку, оскільки саме від цього залежить спектр можливих задач які може виконувати така платформа.

Треба визначити рівень сигналу від одного вузла мережі до іншого, що б гарантувати якісний двосторонній зв’язок. Одним із вузлів виступає плата із мікроконтролером ESP-32, другим вузлом є Android телефон із встановленим додатком терміналу керування.

Для розрахунку дальності бездротового каналу зв’язку потрібно знати деякі величини, такі як: потужність передавача, коефіцієнт підсилення антени передавача, коефіцієнт підсилення антени приймача, чутливість приймача та втрати у антено-фідерному тракті [21].

Дані для ESP-32 взяті із документації до мікроконтролеру [22]. Дізнатися точні величини для телефонів не можливо, через відсутність такої інформації у відкритому доступі, тому будуть взяті середні величини [23]. Втрати антено-фідерного тракту точно дізнатися не можливо, але зазвичай ця величина складає близько 0.5-4 дБ, візьмемо найгірший випадок, тому для обох вузлів будуть втрати 4 дБ. Чутливість приймача візьмемо для швидкості 54 Мбіт/с. Усі дані взяті для стандарту Wi-Fi 802.11b. Отримані величини напишемо у таблицю 3.1

Таблиця 3.1 – Характеристики вузлів

Параметр	**ESP-32**	**Мобільний телефон**
Потужність передавача, [P_t, дБм]	19.5	12
Коефіцієнт підсилення антени, [G, дБм]	2	-2
Чутливість приймача, [P_{min}, дБм]	-88	-86
Втрати антено-фідерного тракту, [L, дБ]	4	4

Для Wi-Fi 802.11b доступно 14 каналів для передачі даних (табл. 3.2), із них в Україні можливо використання перших 13 каналів [24].

Таблиця 3.2 – Частоти каналів 802.11b

Канал	**Частота, МГц**	**Можливе використання**
1	2412	+
2	2417	+
3	2422	+
4	2427	+
5	2432	+
6	2437	+
7	2442	+
8	2447	+
9	2452	+
10	2457	+
11	2462	+
12	2467	+
13	2472	+
14	2484	-

Для того, що б визначити дальність радіозв’язку потрібно знати центральну частоту каналу на якій працює система зв’язку, у нашому випадку, враховуючи усе вище сказане, візьмемо середній 6 канал, це дасть нам загальне представлення про дальність зв’язку в цілому. Отже центральною частотою передачі в нашому випадку буде $F = 2437$ МГц.

Потрібно розрахувати втрати у вільному просторі (FSL – Free Space Loss). Для цього потрібно визначити спільне підсилення усієї системи радіозв’язку за наступною формулою:

$$Y_{\text{дБ}} = P_t + G_t + G_r - P_{min} - L_t - L_r,$$

де P_t – потужність передавача, G_t – коефіцієнт підсилення антени передавача, G_r – коефіцієнт підсилення антени приймача, P_{min} – чутливість приймача,

L_t – Втрати антено-фідерного тракту передавача, L_r – Втрати антено-фідерного тракту приймача.

Для повного розрахунку втрат у вільному просторі, потрібно відняти від них запас по енергії (SOM – System Operating Margin), це запас на усі можливі несприятливі умови для проведення радіозв'язку такі як: сніг, туман, злива, або не налагоджена апаратура чи температурний дрейф приймача і передавача. Загалом вважається, що 10 дБ запасу цілком достатньо для проведення розрахунків. Отже формула розрахунку втрат буде мати наступний вигляд:

$$FSL = Y_{\text{дБ}} - SOM$$

Отримуємо формулу дальності радіозв'язку шляхом перетворень [21] :

$$D = 10^{\left(\frac{FSL}{20} - \frac{32.44}{20} - \log(F)\right)}$$

Проведемо розрахунки дальності зв'язку від ESP-32 до телефону.

1. Загальне підсилення для усієї системи:

$$Y_{ESP} = P_t + G_t + G_r - P_{min} - L_t - L_r =$$
$$= 19.5 + 2 + (-2) - (-86) - 4 - 4 = 97.5 \text{ дБ}$$

2. Втрати у вільному просторі:

$$FSL_{ESP} = Y_{EPS} - SOM = 97.5 - 10 = 87.5 \text{ дБ}$$

3. Дальність каналу зв'язку:

$$D_{ESP} = 10^{\left(\frac{FSL}{20} - \frac{32.44}{20} - \log(F)\right)} = 10^{\left(\frac{87.5}{20} - \frac{32.44}{20} - \log(2437)\right)} = 0.218 \text{ км} = 218 \text{ м}$$

Проведемо розрахунки дальності зв'язку від телефону до ESP-32.

1. Загальне підсилення для усієї системи:

$$Y_{MOB} = P_t + G_t + G_r - P_{min} - L_t - L_r =$$
$$= 12 + (-2) + 2 - (-88) - 4 - 4 = 92 \text{ дБ}$$

2. Втрати у вільному просторі:

$$FSL_{MOB} = Y_{MOB} - SOM = 92 - 10 = 82 \text{ дБ}$$

3. Дальність каналу зв'язку:

$$D_{ESP} = 10^{\left(\frac{FSL}{20} - \frac{32.44}{20} - \log(F)\right)} = 10^{\left(\frac{82}{20} - \frac{32.44}{20} - \log(2437)\right)} = 0.116 \text{ км} = 116 \text{ м}.$$

Даний розрахунок дальності зв’язку передбачає, що приймач та передавач знаходяться на лінії прямої оптичної видимості. Розрахунок не враховує такі фактори як дифракцію та розсіювання радіохвиль.

Із розрахунків виходить, що максимальна теоретична дальність радіозв’язку на стандарті Wi-Fi 802.11b можлива на таких дистанціях:

- Від ESP-32 до мобільного телефону – 218 м.
- Від мобільного телефону до ESP-32 – 116 м.

Оскільки радіозв’язок повинен бути двостороннім, то враховуємо найменшу дальність. Отже дистанція максимального двостороннього радіозв’язку між ESP-32 та мобільним телефоном є 116 м.

3.2.2 Обробка даних із датчиків

Ультразвуковий далекомір US-025 – для початку роботи із ним потрібно подати імпульс логічної одиниці тривалістю 10 мкс на вивід Trig, після чого датчик відправить 8 імпульсів частотою 40 кГц через передавач. Коли імпульси відіб’ються від перешкоди вони надійдуть назад і їх зчитає приймач і відправить імпульс на виводі Echo, потрібно порахувати тривалість даного імпульсу (у мкс) на стороні мікроконтролеру і поділити її на 58.2, що б отримати відстань у см, ця константа залежить від швидкості розповсюдження звуку у середовищі і вибраної одиниці виміру.

Барометр BMP-280 – даний датчик надсилає вже готові дані, їх майже не треба обробляти, лише для визначення висоти над рівнем моря, потрібно вказати атмосферний тиск над рівнем моря.

Тахометр на датчику Холла A3144 – оскільки один прохід магніту над датчиком означає пів оберту ведучого колеса, із цього слідує, що – 2 проходи означають один оберт колеса. Для визначення швидкості достатньо знати діаметр колеса, тобто за одне спрацьовування датчику, гусенична стрічка просувається на половину діаметру ведучого колеса, знаючи це можна визначити відстань, яку пройшла платформа і поділивши її на час – отримати швидкість.

Датчик вологості та температури DHT-11 не потребує особливих розрахунків, оскільки надає уже готові дані. Потрібно лише зауважити, що його опитування не слід проводити з більшою частотою, ніж 1 Гц, оскільки датчик не проводить швидше вимірювання.

Гібридний модуль MPU-6050 являє собою 3-х осьовий гіроскоп і 3-х осьовий акселерометр. Гіроскоп дозволяє отримувати кутову швидкість навколо осі у градусах за секунду. Акселерометр вимірює прискорення вздовж осі у метрах за секунду. Перед проведення вимірів потрібно провести калібрування модулю, що виставляє нульові точки для акселерометру і дрейф гіроскопу, тобто його відхилення у стані покою, для цього потрібно викликати функцію калібрування, вказавши кількість ітерацій калібрування. Для отримування результатів їх необхідно фільтрувати, для цього можна скористатися фільтром Калмана, але найточніший результат дає використання вбудованого DMP (Digital Motion Processor), про те він потребує більшої кількості апаратних ресурсів. Загалом отримання кутів від датчику не має складності при використання готової бібліотеки від розробника.

Датчик заряду батареї – що б дізнатися заряд батареї треба виміряти напругу на ній, звичайно залежність напруги від ємності не лінійна, проте у нашому випадку це не критично. Для вимірювання напруги скористаємося вбудованим у мікроконтролер ATMega328P АЦП. Оскільки діапазон напруги літієвого акумулятору лежить у 2.75 – 4.2В, немає сенсу у дільнику напруги, що б зменшити вхідну напругу. Для вимірювання буде достатньо провести провід від аналогового входу мікроконтролеру до плюсового контакту батареї. АЦП у нашому випадку є 10 розрядним, що надає представлення напруги у від 0 до 1023 значеннях. Опорною напругою буде вхідне живлення, що стабілізуються за рахунок MT3608 і буде дорівнювати 5В. Тобто формула розрахунку напруги буде така:

$$U_{\text{бат}} = (U_{\text{ацп}} * U_{\text{вх}})/1024,$$

де $U_{\text{бат}}$ = напруга батареї; $U_{\text{ацп}}$ = значення АЦП; $U_{\text{вх}}$ = вхідна напруга живлення.

3.2.3 Автономність платформи

Для знаходження часу автономної роботи дистанційно керованої платформи, потрібно визначити споживання струму усіх компонентів системи та порахувати їх суму. За допомогою документації виробників компонентів визначимо максимальне споживання струму усіх компонентів та занесемо їх до таблиці 3.3:

Таблиця 3.3 – Споживання струму компонентів СТРП

Компонент	**Струм, мА**
ESP-32	260
OV2640	40
Arduino Nano	24
DHT-11	3
US-025	6
A3144	<1
BMP280	<1
MPU-6050	5
Індикаторний світлодіод	6
Колекторний двигун * 2	600

Порахуємо максимальний струм навантаження $I_{\text{нав}}$:

$$I_{\text{нав}} = 260 + 40 + 24 + 3 + 6 + 1 + 1 + 5 + 6 + 600 = 946 \text{ мА}$$

Оскільки акумулятори мають середню напругу 3.7В, а уся система так чи інакше живиться від 5В потрібно перевести навантаження у потужність Вт $P_{\text{нав}}$.

$$P_{\text{нав}} = U * I_{\text{нав}} = 5 * 0.946 = 4.73 \text{ Вт}$$

Наявні акумулятори на СТРП мають ємність по 2000 мА*год кожен, їх двоє, отже сумісна ємність 4000 мА*год. Ємність батареї приведемо до Вт * год $\text{С}_{\text{ак}}$:

$$\text{С}_{\text{ак}} = 3.7 * 4 = 14.8 \text{ Вт} * \text{год}$$

Для розрахунку часу автономності скористаємося наступною формулою, ємність акумулятору помножується на 0.93 – це ККД перетворювача напруги:

$$t = \frac{\text{С}_{\text{ак}} * 0.93}{P_{\text{нав}}} = \frac{14.8 * 0.93}{4.73} = 2.9 \text{ год}$$

Отже СТРП повинна працювати 2.9 год при максимальному навантаженні.

3.3 Реалізація додатку для керування платформою

3.3.1 Опис функціоналу додатку керування

Для початку процесу розроблення додатку, потрібно визначитися із його можливостями та функціями, які він повинен забезпечувати. Беручи до уваги вимоги до СТРП (підрозділ 2.1) оглянемо такі функції нижче:

- Підтримувати двосторонній зв’язок із СТРП.
- Надавати можливість вводу команд керування та їх подальшу передачу на СТРП.
- Приймати інформацію із датчиків та виводити їх на екран у зручному для користувача вигляді.
- Надавати можливість підключення інших сервісів, наприклад веб-застосунків для доступу до СТРП.
- Мати можливість запуску на різних операційних системах.

Для кращого наочного сприйняття ролі додатку керування розглянемо схему взаємозв’язків нижче (рис. 3.11).

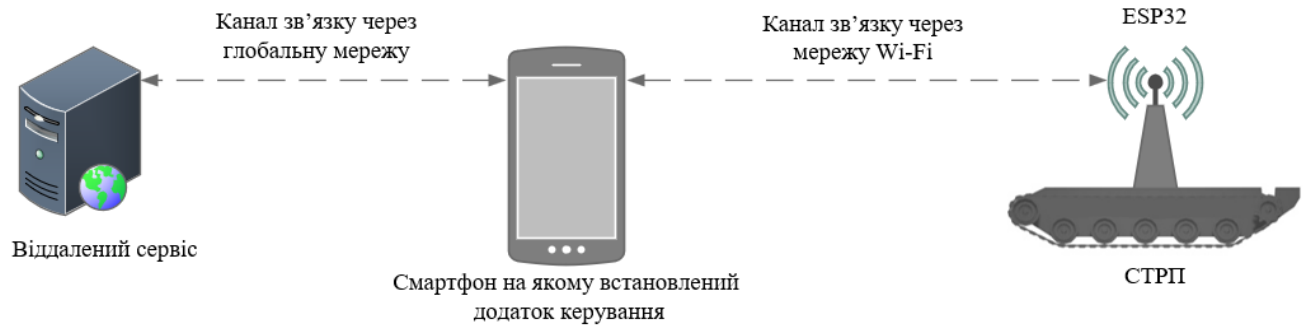


Рисунок 3.11 – Схема взаємозв’язків додатку керування

3.3.2 Методи та засоби розробки додатку керування

Для забезпечення двостороннього зв’язку на смартфоні вмикається точка доступу Wi-Fi мережі, до якої підключається плата із мікроконтролером ESP32. Для організації програмного рівня зв’язку використовується HTTP запити [25], а саме метод GET, оскільки він дозволяє відправити дані та отримати їх у відповідь.

Для надання можливості віддаленого доступу іншим сервісам також доцільно використовувати HTTP запити. Вони є гарним рішенням для реалізації двостороннього зв’язку, оскільки вони по перше дуже популярні у світі, отже мають підтримку на великій кількості різноманітних пристроїв, по друге досить легкі в реалізації, оскільки вже мають готові програмні рішення для розробників і не потребують такої складності реалізації як наприклад передача TCP/UDP пакетів по мережі.

Проте HTTP запити забезпечують лише транспорту функцію, для передачі по запиту інформації потрібно її належним чином підготувати та запакувати у такий собі контейнер. Таким контейнером може виступати JSON [26]. Цей формат дозволяє зберігати різну інформацію у структурованому, текстовому вигляді, що надає можливість передавати у вигляді строк тексту будь-яку інформацію по мережі шляхом HTTP GET запитів.

У якості програмного комплексу, що може стати основою для розробки додатку для більшості операційних систем та комп’ютеризованих платформ, є платформа для розробки інтерактивного контенту Unity [27]. Даний вибір обумовлений відносною легкістю розробки такого роду додатку на даній платформі, її розгортання на багатьох платформах, та подальший супровід додатку керування, оскільки Unity має велику спільноту серед розробників. Беручи до уваги усе вище сказане і те, що СТРП це проект-прототип – Unity є гарним програмним рішенням для розробки додатку керування.

Інструментом для написання програмного коду додатку є – Visual Studio Community 2019 [28]. Основними перевагами даного програмного продукту є те, що він сучасний, потужний і безкоштовний комплекс для розробки майже будь-якого роду програмного забезпечення, до того ж має інтеграцію із Unity. Його користувацьких можливостей цілком достатньо для розробки, налагоджування та подальшого супроводу додатку керування СТРП.

3.3.3 Процес розробки додатку керування

Оскільки охопити розробку під усі платформи одночасно не можливо, то цільовою платформою розробки обрано операційну систему Android, вона встановлена на більшості смартфонів у світі, отже вибір обумовлений її широкою розповсюдженістю, що перекриває більшість можливих ситуацій використання СТРП, а за потреби можливо зібрати додаток для іншої платформи, оскільки програмна платформа Unity це дозволяє. Мовою програмування для розробки додатку є C#.

Опишемо основі етапи створення додатку керування:

- Розгортання програмних засобів: Visual Studio, Unity, Android SDK на ПК.
- Створення проекту у Unity та Visual Studio для ОС Android.
- Розробка інтерфейсу користувача, який повинен забезпечувати управління СТРП, та відображення отриманих із неї даних.
- Створення частини коду, що відповідає за прийняття показників датчиків.
- Створення частини коду, що відповідає за прийняття, декодування та відображення вбудованої у СТРП камери.
- Створення частини коду, що відповідає за прийняття, обробку та подальше передавання доступу до даних можливим сервісам, через HTTP GET запити.

Загалом додаток керування є своєрідним містком між СТРП та іншими можливими віддаленими сервісами, тобто виконує роль програмного шлюзу. Отже розроблений термінал керування для самохідної телеметричної радіокерованої платформи у вигляді додатку для смартфону виконує добре, маючи при цьому багатий потенціал подальшого покращення та впровадження нового функціоналу, за рахунок використання програмних засобів, що використовувались у процесі розробки додатку.

ВИСНОВОК

У третьому розділі описаний процес створення СТРП, а саме: підготовка шасі, монтаж усіх модулів датчиків, радіодеталей та допоміжних компонентів. Для кожного датчику було детально розібрані схеми підключення, беручи до уваги їх конкретні властивості та монтаж на СТРП. У процесі налаштування роботи датчиків були записані особливості роботи із ними.

Також описано принцип роботи драйверу для двигунів та на основі цього створена схема підключення їх до драйверу і керуючого мікроконтролеру.

Беручи до уваги потребу у зв'язку між двома мікроконтролерами у складі СТРП було створено схему підключення UART інтерфейсу між ними.

Оскільки одним із основних характеристик дистанційно керованої техніки є дальність її ефективного використання було проведено розрахунок дальності зв'язку. Із даного розрахунку було виведеного, що на дистанції до 100 м прямої видимості, СТРП має можливість дистанційного керування.

Для СТРП було порахований теоретичний час автономної роботи від акумуляторної батареї – він склав 2.9 год.

Наприкінці був описаний процес створення додатку для мобільного телефону, що б керувати СТРП, у ході якого були визначені можливості додатку, як терміналу керування та основні принципи, що використовувались для його розробки.

ЗАГАЛЬНІ ВИСНОВКИ

Дана кваліфікаційна робота мала перед собою мету розроблення самохідної, телеметричної, радіокерованої системи та додатку керування для неї.

У першому розділі було проведено детальний аналіз сучасних дистанційно керованих систем на прикладі різноманітної техніки. Отримані у ході аналізу дані було використано для класифікації основних видів віддалено керованих апаратів. На основі чого було сформовано принципи побудови телеметричної радіокерованої системи.

У другому розділі були описані функціональні вимоги до СТРП. За допомогою опису платформи було сформовано список компонентів необхідних для побудови дистанційно керованої платформи. Був проведений огляд сучасного асортименту можливих компонентів та їх характеристик. На основі огляду було вибрано та обґрунтовано вибір двох плат із мікроконтролерами, одна з них виконує роль керування, інша для в якості комунікаційного блоку, окрім цього було обрано датчики, відеокамеру та інше необхідне обладнання для ефективного виконання поставлених перед СТРП завдань.

У третьому розділі був описаний процес збирання, монтажу та під’єднання усіх компонентів СТРП у єдину систему. Описані особливості налаштування обладнання та організація зв’язку між двома мікроконтролерами. Для усіх електричних з’єднань були побудовані принципові схеми. Проведено розрахунки основних параметрів дистанційно керованої платформи – автономності та дальності Wi-Fi зв'язку. Для віддаленого керування та збирання показань із датчиків і відображення відео, було розроблено термінал керування у вигляді мобільного додатку для ОС Android.

Отже у ході даної кваліфікаційної роботи було виконано поставлені початкові завдання, а саме – аналіз сучасних дистанційних систем, на основі чого розроблено і побудовано самохідну телеметричну радіокеровану систему із додатком керування для неї.

СПИСОК ДЖЕРЕЛ І ЛІТЕРАТУРИ

1. Lichiardopol, S. (2007). A survey on teleoperation. (DCT rapporten; Vol. 2007.155). Technische Universiteit Eindhoven.
2. Основи робототехніки/ Fundamentals of robotics: навч. посіб. / Н. В. Морзе, Л. О. Варченко-Троценко, М. А. Гладун ; Київ. ун-т ім. Бориса Грінченка. — Кам'янець-Подільський (Хмельниц. обл.): Буйницький О. А., 2016. —183 с
3. What are remote controlled robots? How to build a remote controlled robot? [Електронний ресурс] – Режим доступу до ресурсу: https://www.skyfilabs.com/blog/what-are-remote-controlled-robots-and-how-to-build.
4. William Murdoch [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/William_Murdoch.
5. Пильчиков Микола Дмитрович [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Пильчиков_Микола_Дмитрович.
6. Советские роботы на ЧАЕС [Електронний ресурс] – Режим доступу до ресурсу: https://www.forbes.ru/tehnologii/377515-o-kom-zabyli-v-seriale-chernobyl-sovetskie-roboty-na-avarii-aes.
7. История советской робототехники [Електронний ресурс] – Режим доступу до ресурсу: https://statehistory.ru/4498/Istoriya-sovetskoy-robototekhniki/.
8. Industrial robot [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Industrial_robot.
9. Mars Exploration Rovers [Електронний ресурс] – Режим доступу до ресурсу: https://mars.nasa.gov/mer/.
10. Perseverance's Selfie with Ingenuity [Електронний ресурс] – Режим доступу до ресурсу: https://mars.nasa.gov/resources/25790/perseverances-selfie-with-ingenuity/

11. Беспилотный летательный аппарат [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/Беспилотный_летательный_аппарат.
12. Дистанционно управляемые аппараты (ROV) [Електронний ресурс] – Режим доступу до ресурсу: https://maritime-zone.com/news/view/2009-04-16-rovs.
13. Корсунский В. А. Дистанционно управляемые инженерные и дорожные машины / В. А. Корсунский. // Журнал Дорожные машины.
14. Lichiardopol, S. (2007). A survey on teleoperation. (DCT rapporten; Vol. 2007.155). Technische Universiteit Eindhoven.
15. Мобильная Робототехника [Електронний ресурс] – Режим доступу до ресурсу: https://academy.evolvector.ru/robototehnika/mob-robototeh/.
16. Jónatas R. Development of a structure for a mobile robot / Jónatas Rodrigues – Institute of Systems and Robotic of the University, 2017.
17. Васильев А. В. Принципы построения и классификация шасси мобильных роботов наземного применения и планетоходов / А. В. Васильев. // Журнал Информатика. Телекоммуникации. Управление. – 2013.
18. Датчик [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Датчик.
19. Виконавчий механізм [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Виконавчий_механізм.
20. Мікроконтролер [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Мікроконтролерhttps://uk.wikipedia.org/wiki/Мікроконтролер.
21. Free-space path loss [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Free-space_path_loss.
22. ESP32 Series Datasheet [Електронний ресурс] – Режим доступу до ресурсу: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.
23. WIFI Antenna Design [Електронний ресурс] – Режим доступу до ресурсу: https://www.antenna-theory.com/design/wifi.php.

24. Смуги радіочастот Wi-Fi [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Смуги_радіочастот_Wi-Fi.
25. Методы HTTP запроса [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Web/HTTP/Methods
26. Работа с JSON [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Learn/JavaScript/Objects/JSON
27. Unity Real-Time Development Platform [Електронний ресурс] – Режим доступу до ресурсу: https://unity.com/
28. Visual Studio 2019 [Електронний ресурс] – Режим доступу до ресурсу: https://visualstudio.microsoft.com/vs/

Додаток А

Переклад першого розділу

1. THEORY AND GENERAL INFORMATION ABOUT REMOTE CONTROLLED VEHICLES

1.1 History of robotics and radio-controlled models

1.1.1 Remotely controlled game models

Great development work managed have given a model remote-controlled toy, so we need to consider this event in more detail.

The first radio-controlled toys became possible to appear glow plug engine, which at small sizes provided sufficient power for small models. The first models had no control and ran in a circle. However, control at that time was only in the form of turning the engine on and off, the development of full-fledged control with the front wheels and engine power, appeared only in the 60's.

In 1784 british inventor William Murdoch working to improve the steam engine proposed a model of self-propelled carriages to steam course that can carry up to 5 tons - it was one of the first machine capable of independent movement, however, without control [6].

1.1.2 Remotely controlled vehicles in the military sphere

Military conflicts in human history are the engines of progress, so it is necessary to analyze in more detail the technologies used at that time.

In the 1930s, the Soviet Union conducted tests on the MS-1 tank with installed remote control.

During the examinations, the expediency of developing such radio-controlled tanks was determined. In 1933, a specialized tank TT-18 appeared, in which control was full remote. The tank control system could accommodate 16 commands, including: change of speed, turns of the hull, engine stop and detonation of explosives. The true maximum range of 500-1000 m was the tank as a whole.

The tank as a whole turned out to be a successful project (as a radio tank), it had easy control and good passability, but due to the design features of the tank base, it returned when driving on potholes, so it did not go into the series.

In 1940 on the base of the tank BT-7 developed a new teletank, it already had autonomous of 4-6 hours, range control up to 4 km and 17 command control system. The control itself seemed successful, but as a combat vehicle it had poor accuracy, so it was abandoned [11].

1.1.3 Robots for extreme conditions

Also, Soviet scientists after the catastrophic event at the Chernobyl nuclear power plant in 1986 developed mobile works for work in the work area. In total, there were about 15 robots, among them were models such as: caterpillar "Mobot-CHHV", robot "STR-1" and wheeled robot scout "RR-1".

Light robots measured the radiation situation and transmitted telemetry data to visualize and control the working area for other robots. Heavier works were assigned to carry out technological works on cleaning and decontamination of future construction sites.

In total, the robots replaced the work of hundreds of people without exposing them to danger, because in some places the radiation level was 18,000 R/h [1, 2]. An example of one of these robots can be seen in the figure (Fig. 1.1).

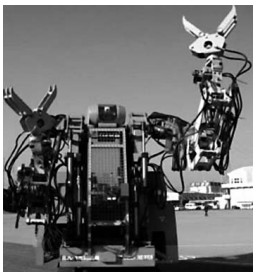


Figure 1.1 – Japanese emergency mobile robot T-52

1.2 Analysis modern state of robotics and its prospects

Modern robots with a remote control can be divided into the following type of use [3, 8]:

- Spacecraft, most of which have remote control capabilities on board.
- Military robots are widely used for reconnaissance and strikes on the enemy.

- Remote manipulators for remote operation by the operator with hazardous materials, such as radioactive or explosive.
- Techniques for scientific purposes include both spacecraft on the surface of other planets and submarines for various studies, where human presence is not possible or economically impractical.
- Industrial manipulators for various works - welding, loading, cutting, painting.

1.2.1 Space programs for the use of remote vehicles

Open space is a huge space for the operation of robots with remote control, because finding a person there is not economically impractical, and in most cases, even dangerous or impossible.

Space objects with remote control in space can be divided into the following groups:

- Artificial satellites - is specialized devices used to relay signals, weather forecasts or work positioning systems (GPS, GLONASS).
- Scientific works on space exploration are various space stations, scout probes, observers, landing robots, all of which require advanced remote-control systems.
- Military space objects - there is not much information about them, it is known that they are remotely controlled.

Because most famous remotely managed objects in the solar second system devices are created in the space program NASA « Mars Exploration Rover » (MER), so consider them more [16].

During the MER program, several rovers were created, so consider their common features.

The goals set for these vehicles were:

- Studied Martian soil and rocks that were used to collect samples manipulators and systems analysis.
- Search for valuable iron-bearing minerals, for this purpose used cameras with different light filters.

- Search for water and minerals that could indicate its presence.
- Assessing the quality of life in the future on Mars.

The rover itself was a vehicle on wheels based and weighing about 200 kg, several cameras, spectrometers, a manipulator with built-in sensors and a micro-camera.

Martian spacecraft are hybrid remote-controlled vehicle. They receive a command to move from the operator, but how exactly to move on which trajectory they decide for themselves, by building a stereoscopic map of the environment, from which they understand where the obstacles are and bypass them. They analyze obstacles according to the criteria of obstacle height, soil density, surface angle and choose the safest and shortest way from dozens of possible paths. The spacecraft can be seen in the figure below (Fig. 1.2) [29].

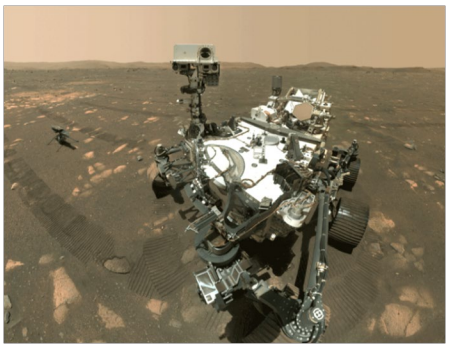


Figure 1.2 – Mars rover «Perseverance»

1.2.2 Military developments in the field of remotely controlled vehicles

As in the space industry, military programs require reconnaissance vehicles, because information about the enemy is very valuable. Remotely controlled reconnaissance vehicles are the most common in military affairs. A striking example of this is the UAV - an unmanned aerial vehicle. They can be of varying degrees of autonomy, from manually operated by the operator, to almost completely automatic. In general, their main advantage is a significantly lower cost compared to a traditional aircraft and no risk of human loss [14].

The UAV system consists of:

- The drone itself;
- Communication system: it can be radio or satellite communication;

- Operator control point;
- Additional equipment is required for transportation or maintenance of UAV.

A special type of military equipment is robot sappers, they perform demining of explosive devices. They are usually structurally composed of a tracked platform, several chambers and manipulators, and can also be equipped with water / foam guns.

A striking example of such a technique is UAV "Phantom" (Fig. 1.3).

Figure 1.3 - Ukrainian multi-purpose tactical device "Phantom"

1.2.3 Remote control submarines

In the case of remotely operated underwater vehicle (remotely operated vehicles, ROV), there is no more economically feasible and safe way to perform underwater manipulations.

ROV devices differ significantly in size. From those that hold only a television camera and are used only for surveillance, to systems that have mechanical devices, a few flexible manipulators and other equipment.

The most common class are working machines that have a fairly large power (50-100 hp). They can carry up to 300 kg of cargo, some even up to 500 kg. Typical tasks of this class of vehicles are light underwater construction, pipe research or drilling [17].

Today, such devices can perform a range of tasks such as: research, engineering transportation or installation, search and rescue operations.

1.2.4 Engineering remote vehicles

Given the fact that the general level of technology already allows the construction of almost autonomous self-propelled platforms, this could not help but be used in engineering. The machines itself is a vehicle on which the appropriate tools, devices and sensors are installed to perform a cycle of technological operations.

Mobile engineering platforms perform a number of works in the oil, mining, gas and civil engineering regions.

The main goals of engineering mobile platforms are [18]:

- Reducing the number of staff;
- Carrying out works in the conditions harmful or dangerous for the person;
- Increasing mobility and efficiency in mining, excavation and other works;
- Reduction of fatigue and occupational injuries in workers.

Modern applications of mobile platforms are search and rescue operations in case of natural disasters, mining operations for resource extraction, logging operations, etc. An example of this technique is also engineering robotic complex RTS-U (Fig. 1.4).

Fig. 1.4 - I engineering complex RTS-U

1.2.5 Medical system for remote control

Thanks to the development of robotic technology, a revolution in the medical field began. For example, the development of imaging devices has made it possible to make more accurate diagnoses and conduct less invasive treatments. In general, the topic of remotely controlled devices is suitable for robotic surgery, an area with great prospects.

Robotic surgery has been established with the aim to empower surgeons, who perform surgery [13]. The surgeon has the opportunity to perform the operation in two ways:

- Manually controlling the remote manipulator with cameras, representing robotic hands;
- With the help of computer control, where the surgeon only gives instructions on how to perform the operation, this method is good because it may not require the presence of a surgeon.

Thanks to the achievement of robotic surgery, operations are performed with greater accuracy, less intervention in the human body and have less blood loss. In addition, surgeons have more control over the area of operation, less fatigue because they do not need to stand over the patient all the time, and the natural trembling of the fingers is filtered by the robot software. A popular solution for such remote medical equipment is a robotic surgical system "da Vinci" (Fig. 1.5).

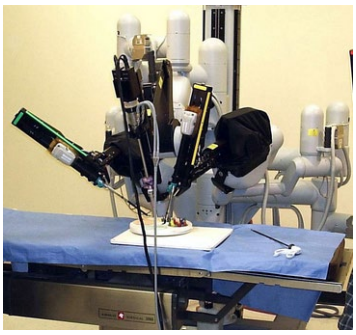


Fig. 1.5 – Surgical system "da Vinci"

1.3 General principles of building a radio-controlled model

1.3.1 Main components

To effectively build remotely controlled robots, you need to understand the basic components of which they consist. In general, they include such components as: casing, chassis, power supply, sensors, executive mechanism, control unit, transceiver and software [19].

The components of the radio-controlled model and their interaction can be seen more clearly in the figure below (Fig. 1.6).

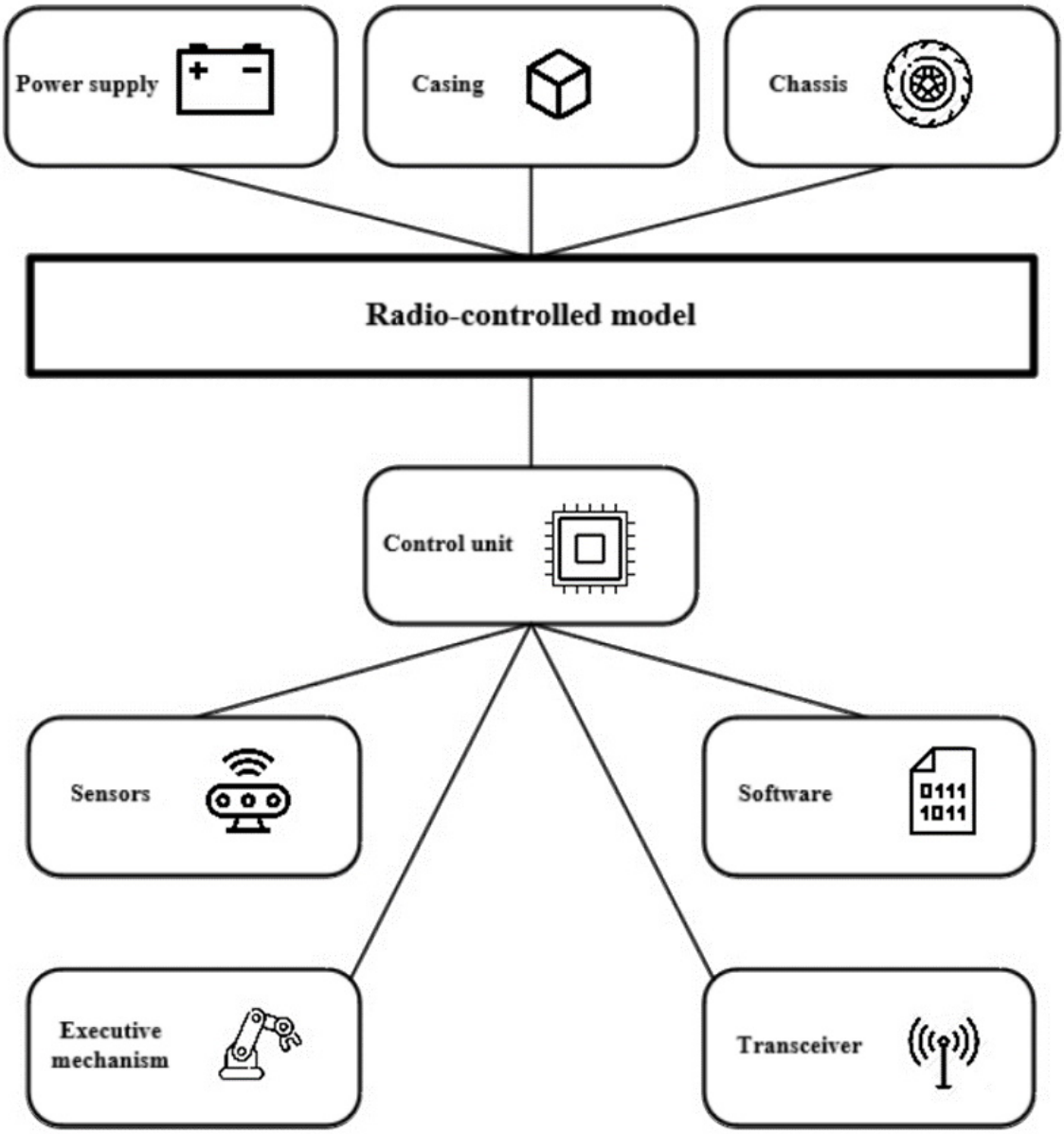


Fig. 1.6 – Functional diagram of the radio-controlled model

1.3.2 Casing

The basis of any autonomous, remotely controlled equipment is - the casing. It provides the basis for further construction of the robot. Its choice depends on the characteristics of a system [20].

The following materials can be used for the casing:

- Aluminum is a light metal that provides strength and is sufficiently resistant to corrosion.
- Acrylic - is popular because of its beautiful transparent appearance.

- Plywood is quite heavy and can have problems with moisture, but it is very easy to handle.
- Polycaprolactone - a plastic with a low melting point, is usually used for prototyping.
- Different types of plastic - the middle ground between strength, weight and complexity of processing.
- Steel - provides great strength and durability, but has a lot of weight, is used in important technology.

Depending on the purpose of the remote-controlled vehicles as a casing, choose if it is a hobby project: a variety of ready-made casing from other vehicles or toys, printing on a 3D printer, developing your own plastic casing, etc. If it is an industrial robot, then casing made to order from sheet metal or plastic or printed on a 3D printer are usually used.

1.3.3 Chassis

The chassis of any mobile platform, more than other components is responsible for its type: wheeled vehicles, tracked or possibly some exotic screw-propelled propulsion [23]. Next, we will consider the currently popular chassis options.

Wheeled - a very common version of the chassis for mobile systems, characterized by its ease of creation and development, have a medium passability, the fastest of the ground versions of the chassis. Of the disadvantages is the turning radius, but in some special wheels such a disadvantage is absent (Ilona wheel).

Walking is a variety of structures that bend on hinges. The movement is carried out due to the synchronous movement of the supporting structures. The advantage is a fairly high passability. Disadvantages: the complexity of development and creation of not only design but also software.

Tracked - have a large passability, small turning radius, the ability to turn on the spot, high resistance to overturning. Disadvantages: difficult to create a structure, especially if there is a suspension, require frequent technical inspection.

Hybrid is a type of chassis that can include all types, such as tracks at the rear of the car and wheels at the front. They combine the disadvantages and advantages of all types of chassis that are part of them, so the review of such chassis is possible only by a specific example.

1.3.4 Power supply

The power supply unit is a functional unit of a self-propelled system that provides electricity to other units of the machine. In others may be missing or replaced with an adapter, if power is supplied through cable is often the case in underwater or self-military propelled systems. The most popular solution is when the power supply is batteries or accumulators. From the power supply depends on engine power, and therefore permeability self-propelled system, but first power supply is - time of autonomy of all system.

1.3.5 Sensors

A sensor is a device whose main purpose is to translate the measurements of physical quantities around a self-propelled system into electrical signals for further processing.

Sensors are a bridge between the environment and the robot [21].

There are a very large number of sensors, so we list the basic data that can be obtained with them:

- Lighting level - photodiodes;
- Distance measurement - laser, ultrasonic ranging module;
- Level of acceleration on several axes - accelerometers;
- Level of inclination - gyroscopes;
- Humidity / rain level - resistive sensors;
- Sound level - microphones;
- Magnetic field level - magnetometers;
- Temperature and humidity level - thermometers and hygrometer;
- Voltage and current level - analog-to-digital converters (ADC);

- Complicated modular sensors - GPS, line sensors, gestures, voice commands, ionizing radiation, etc.

Some of the sensors are modules that allow the measurement of several physical quantities and process the raw data, which allows us to get from some quantities to others in analog or digital form.

1.3.6 Executive mechanism

The executive mechanism is a device by which a self-propelled system interacts with the environment, it can be a lantern or a manipulator to move objects. Usually such a mechanism includes electric motors, but there are other drives such as pneumatic drives [22].

1.3.7 Control unit

The control unit is one or more electronic boards that can contain microcontrollers, they process the information coming from the sensors and make appropriate decisions based on them, for example, execute the executive mechanism.

In addition, the control unit receives remote commands from the operator and can control other functional units of the self-propelled system.

There can be several control units, each of which performs a specific function or entered for redundancy, in which case they are all combined under the direction of a central control unit.

1.3.8 Transceiver

Transceiver - is a device that can receive and transmit messages through the environment, it can be radio waves, laser or light radiation, or electromagnetic waves over a cable. One of its characteristics is the distance and speed of information transmission, so the possibility of action scenarios for the mobile remote-control system. Its purpose in the self-propelled system is to transmit telemetry information received by the system from sensors and receive control signals from the operator.

1.3.9 Software

Software is a control program or programs for a self-propelled system, which includes algorithms for processing data from sensors, their calibration, analysis, conversion for convenient form. The software also describes algorithms for processing commands from the operator, so that the executive mechanism would work as expected.

In modern world and the development of technology, software is one of the most valuable components of a self-propelled telemetry system, because the program depends on how it will perform its tasks.

CONCLUSION

In the first chapter was examined constructs in remote control systems of the example of toy models, military and engineering vehicles, was analyzed modern state of robotics, namely remotely controlled systems and their classification.

During the classification, the main areas of remote-controlled vehicles were identified, such as spacecraft, military, scientific vehicles and manipulators for industrial needs, in addition, there is also medical system with telemetry control. All classes of this technique have certain requirements specific to their use and they vary greatly depending on the purpose of the vehicles.

After reviewing the history and modern state of robotics, the basic principles of building a self-propelled telemetry radio-controlled platform were formed. The principles of construction of STRP components were defined such as: casing, chassis, power supply, sensor unit, executive mechanism, control unit and transceiver.

Thanks to the analysis of classification and principles of construction of remotely controlled systems, it is possible to begin work on finding hardware in the next section.

Додаток Б

Лістинг коду додатку керування

Файл HTTP_Manager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Networking;


public class HTTP_Manager : MonoBehaviour
{
    public Data data_update;
    public RemoteServer r_server;

    private bool is_debug = false;
    private string strp_url = "http://192.168.1.103:1053/";
    private string data_patch = "?data=true";
    private string control_patch = "?control=";

    public void GetData()
    {
        StartCoroutine(SendDataGetRequest());
    }

    public void SendControl(string payload)
    {
        StartCoroutine(SendControlGetRequest(payload));
    }

    private IEnumerator SendDataGetRequest()
    {
        string url = strp_url + data_patch;

        float time = Time.unscaledTime;
        UnityWebRequest uwr = UnityWebRequest.Get(url);
        yield return uwr.SendWebRequest();

        if (uwr.result == UnityWebRequest.Result.ConnectionError)
        {
            if (is_debug)
                Debug.Log("Error DATA req: " + uwr.error);
        }
        else
        {
            string data = uwr.downloadHandler.text;
            if (is_debug)
            {
                Debug.Log("Received: " + data);
```

```
                Debug.Log("Ping DATA = " + ((Time.unscaledTime - time) *
1000).ToString());
            }
            data_update.TextsUpdate(data);
            r_server.SendData(data);
        }
    }

    private IEnumerator SendControlGetRequest(string payload)
    {
        string url = strp_url + control_patch + payload;

        float time = Time.unscaledTime;
        UnityWebRequest uwr = UnityWebRequest.Get(url);
        yield return uwr.SendWebRequest();

        if (uwr.result == UnityWebRequest.Result.ConnectionError)
        {
            if (is_debug)
                Debug.Log("Error Control req: " + uwr.error);
        }
        else
        {
            if (is_debug)
            {
                string data = uwr.downloadHandler.text;
                Debug.Log("Received: " + data);
                Debug.Log("Ping Control = " + ((Time.unscaledTime - time) *
1000).ToString());
            }
        }
    }
}
```

Файл Data.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TinyJson;
using UnityEngine.UI;

public class Data : MonoBehaviour
{
    public HTTP_Manager http_manager;
    public Text battery_t;
    public Text RSSI_t;
    public Text temp_t;
    public Text hum_t;
    public Text ultrasonic_t;
```

```
    public Text speed_t;
    public Text mileage_t;
    public Text acc_t;
    public Text gyro_t;
    public Text angle_t;
    public Text pressure_t;
    public Text altitude_t;

    private bool data_is_update = true;
    private const float data_update_time = 1;

    private float timer_data_update = 0;
    Dictionary<string, object> data_endcoded;

    private void Start()
    {
        timer_data_update = Time.unscaledTime + data_update_time;
    }

    private void Update()
    {
        if (Time.unscaledTime > timer_data_update && data_is_update)
        {
            timer_data_update = Time.unscaledTime + data_update_time;

            http_manager.GetData();
        }
    }

    public void TextsUpdate(string data)
    {
        if (data != null && data.Length > 0)
        {
            data_endcoded = (Dictionary<string,
object>)data.FromJson<object>();

            battery_t.text = "Батарея: " + data_endcoded["bat"].ToString()
+ " В / " + data_endcoded["bpc"].ToString() + " %";
            RSSI_t.text = "RSSI:" + data_endcoded["sig"].ToString() + "
дБм";
            temp_t.text = "Температура: " + data_endcoded["tmp"].ToString()
+ " °C";
            hum_t.text = "Вологість: " + data_endcoded["hum"].ToString() +
" %";
            ultrasonic_t.text = "Перешкода: " +
data_endcoded["u_dst"].ToString() + " см";
            speed_t.text = "Швидкість: " + data_endcoded["spd"].ToString()
+ " см/с";
            mileage_t.text = "Пробіг: " + data_endcoded["mil"].ToString() +
" см";
```

```
            pressure_t.text = "Тиск: " + data_endcoded["prs"].ToString() +
" гПа";
            altitude_t.text = "Висота: " + data_endcoded["alt"].ToString()
+ " м";
            acc_t.text = "Акселерометр: [" +
((List<object>)data_endcoded["acc"])[0] + ';' +
((List<object>)data_endcoded["acc"])[1] + ';' +
((List<object>)data_endcoded["acc"])[2] + "] м/с";
            gyro_t.text = "Гіроскоп: [" +
((List<object>)data_endcoded["gyr"])[0] + ';' +
((List<object>)data_endcoded["gyr"])[1] + ';' +
((List<object>)data_endcoded["gyr"])[2] + "] °/с";
            angle_t.text = "Кут нахилу: [" +
((List<object>)data_endcoded["ang"])[0] + ';' +
((List<object>)data_endcoded["ang"])[1] + ';' +
((List<object>)data_endcoded["ang"])[2] + "] °";
        }
    }
}
```

Файл Control.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TinyJson;

public class Control : MonoBehaviour
{
    public HTTP_Manager http_manager;

    private float turn_off_timeout = 0.25f; //sec

    private Dictionary<string, int> control_dic = new Dictionary<string,
int>();
    private float timer_turn_off = 0;
    bool is_turn = false;
    private enum MoveSide
    {
        FORWARD,//0
        BACK,   //1
        LEFT,   //2
        RIGHT,  //3
        STOP    //4
    }

    private void Update()
    {
        if (is_turn && Time.unscaledTime > timer_turn_off)
        {
            is_turn = false;
```

```
            STRP_Move((int)MoveSide.STOP);
        }
        KeyboardControl();
    }

    public void STRP_Move(int side)
    {
        MoveSide move_side = (MoveSide)side;

        control_dic["control"] = (int)move_side;
        string payload = control_dic.ToJson(); // {"control":N}

        http_manager.SendControl(payload);

        if (move_side == MoveSide.LEFT || move_side == MoveSide.RIGHT)
        {
            is_turn = true;
            timer_turn_off = Time.unscaledTime + turn_off_timeout;
        }
    }

    public void KeyboardControl()
    {
        if (Input.GetKeyDown(KeyCode.UpArrow))
        {
            STRP_Move((int)MoveSide.FORWARD);
        }
        if (Input.GetKeyDown(KeyCode.DownArrow))
        {
            STRP_Move((int)MoveSide.BACK);
        }
        if (Input.GetKeyDown(KeyCode.LeftArrow))
        {
            STRP_Move((int)MoveSide.LEFT);
        }
        if (Input.GetKeyDown(KeyCode.RightArrow))
        {
            STRP_Move((int)MoveSide.RIGHT);
        }
        if (Input.GetKeyDown(KeyCode.Keypad0))
        {
            STRP_Move((int)MoveSide.STOP);
        }
    }
}
```

Файл RemoteServer.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Networking;
using TinyJson;
public class RemoteServer : MonoBehaviour
{
    public Control control;

    private bool remote_data = true;
    private bool remote_control = true;
    private bool is_debug = false;

    private string server_data_patch =
"http://31.131.23.133/lib/sensors.php?data=";
    private string server_control_patch =
"http://31.131.23.133/lib/command.php?control=get";
    private const float control_update_time = 0.1f;
    private float timer_control_update = 0;
    private void Start()
    {
        timer_control_update = Time.unscaledTime + control_update_time;
    }
    private void Update()
    {
        GetControl();
    }
    public void GetControl()
    {
        if (remote_control && Time.unscaledTime > timer_control_update)
        {
            timer_control_update = Time.unscaledTime + control_update_time;
            StartCoroutine(GetControlFromRemoteServer());
        }
    }
    public void SendData(string payload)
    {
        if (remote_data)
            StartCoroutine(SendDataToRemoveServer(payload));
    }
    private IEnumerator SendDataToRemoveServer(string payload)
    {
        string url = server_data_patch + payload;

        float time = Time.unscaledTime;
        UnityWebRequest uwr = UnityWebRequest.Get(url);
        yield return uwr.SendWebRequest();

        if (uwr.result == UnityWebRequest.Result.ConnectionError)
```

```
        {
            if (is_debug)
                Debug.Log("Error Remote Server DATA req: " + uwr.error);
        }
        else
        {
            if (is_debug)
            {
                string data = uwr.downloadHandler.text;
                Debug.Log("Received: " + data);
                Debug.Log("Ping Remote Server Data = " +
((Time.unscaledTime - time) * 1000).ToString());
            }
        }
    }
    private IEnumerator GetControlFromRemoteServer()
    {
        string url = server_control_patch;

        float time = Time.unscaledTime;
        UnityWebRequest uwr = UnityWebRequest.Get(url);
        yield return uwr.SendWebRequest();

        if (uwr.result == UnityWebRequest.Result.ConnectionError)
        {
            if (is_debug)
                Debug.Log("Error Remote Server DATA req: " + uwr.error);
        }
        else
        {
            string data = uwr.downloadHandler.text;
            if (is_debug)
            {
                Debug.Log("Received: " + data);
                Debug.Log("Ping Remote Server Control = " +
((Time.unscaledTime - time) * 1000).ToString());
            }
            Debug.Log("Received: " + data);
            Dictionary<string, object> data_json = (Dictionary<string,
object>)data.FromJson<object>();

            int side = (int)data_json["control"];
            if (side == 5)
            {
                side = 0;
            }
            control.STRP_Move(side);
        }
    }
}
```

Додаток В

Лістинг програмного коду ESP-32-S

Файл ESP_32_ROM.ino

```
#include <esp_camera.h>
#include <esp_int_wdt.h>
#include <esp_task_wdt.h>
#include <WiFi.h>
#include <DNSServer.h>
#include <ArduinoOTA.h>
#include "src/parsebytes.h"
struct station { const char ssid[65]; const char password[65]; const bool
dhcp;};
#include "myconfig.h"
#include "src/version.h"
#include "camera_pins.h"
#include "storage.h"
#include "additional.h"

#include <ArduinoJson.h>
#include <HTTPClient.h>

//const
#define SERIAL_SPEED                        115200
#define SERIAL_TIMEOUT                           200
#define JSON_DATA_IN_SIZE                   500
#define JSON_DATA_DETERMINATOR              '$'
#define JSON_DATA_TERMINATOR                '#'
#define HTTP_SERVER_PORT                    1053
#define HTTP_SERVER_MAX_REQUEST             500
const String HTTP_SERVER_DATA_ARG = "data";
const String HTTP_SERVER_CONTROL_ARG = "control";

//vars
unsigned long timer_send_GET_data = 0;
StaticJsonDocument<JSON_DATA_IN_SIZE> data_in;
WiFiServer http_server(HTTP_SERVER_PORT);
WiFiClient http_client;

bool data_is_init = false;

void setup()
{
    Serial.begin(SERIAL_SPEED);
    Init();
     http_server.begin();
}

void loop()
```

```
{
      AdditionalUpdate();
      ReceiveData();
      Update_HTTP_Server();
}

void ReceiveData()
{
      if(Serial.available())
      {
            if (Serial.read() == JSON_DATA_DETERMINATOR)
            {
                  Serial.flush();//wait for end transmition
                  String bufer =
Serial.readStringUntil(JSON_DATA_TERMINATOR);

                  Serial.println();
                  Serial.println(bufer);
                  Serial.println();

                  DeserializationError err = deserializeJson(data_in,
bufer);
                  data_in["sig"] = WiFi.RSSI();//add signal level

                  if (!data_is_init)
                  {
                        data_is_init = true;
                  }

                  if (err == DeserializationError::Ok)
                  {
                        //Serial.println("JSON INPUT\n");
                        //serializeJson(data_in, Serial);
                        //Serial.println();
                        //Send_GET_Data();
                  }
                  else
                  {
                        Serial.println("DeserializationError Code = " +
String(err.c_str()));
                  }
            }
    }
}
void Update_HTTP_Server()
{
      http_client = http_server.available();
    if (http_client)
    {
      bool arg_write = false;
        bool value_write = false;
```

```
  bool end_write = false;
  unsigned int i = 0;
  String argument = "";
  String value = "";
  String answer = "";
while (http_client.connected())
{
   if (http_client.available())
   {
      char c = http_client.read();
      Serial.write(c);

      if (arg_write)
      {
        argument += c;
      }
          if (value_write)
          {
                if (c == ' ')
                {
                      end_write = true;
                }
                else
                {
                      value += c;
                }
          }
      if (c == '?')
      {
                arg_write = true;
      }
          else if (c == '=')
          {
                value_write = true;
                arg_write = false;
                argument.remove(argument.length() - 1); //remove '='
          }
      else if (c == '\n' || end_write)//end
      {
                if (argument == HTTP_SERVER_DATA_ARG)
                {
                      if (data_is_init)
                      {
                            serializeJson(data_in, answer);
                      }
                }
                else if (argument == HTTP_SERVER_CONTROL_ARG)
                {
                      Serial.println();
                      Serial.print(JSON_DATA_DETERMINATOR);
                      value.replace("%7B", "{");
```

```
                        value.replace("%22", "\"");
                        value.replace("%7D", "}");
                        Serial.print(value);
                        Serial.print(JSON_DATA_TERMINATOR);
                        Serial.println();
                    }
                http_client.println("HTTP/1.1 200 OK");
                http_client.println("Content-Type:text/plane");
                http_client.println("Connection: close");
                http_client.println();
                http_client.println(answer);
                http_client.println("");
                break;
            }
        }
            if (i > HTTP_SERVER_MAX_REQUEST)
            {
                Serial.println("Error, max http request");
                break;
            }
            i++;
      }
      http_client.stop();
    }
}
```

Додаток Г

Лістинг програмного коду MEGA 328P (Arduino Nano)

Файл NANO_ROM.ino

```
//Lib
#include "DHT.h"
#include "NewPing.h"
#include "Wire.h"
#include "MPU6050_light.h"
#include <Adafruit_BMP280.h>
#include <ArduinoJson.h>
#include "PinChangeInterrupt.h"
//debug
#define SENSORS_SERIAL                          false
#define SEND_DATA                               true
//const
#define SERIAL_SPEED                        115200
#define SERIAL_TIMEOUT                          200
#define FAST_SENSOR_UPDATE_TIME             500 //for fast sensors
#define SLOW_SENSOR_UPDATE_TIME       2000 //for slow sensors
#define SENSOR_SERIAL_PRINT_TIME            1000
#define MAX_US025_DISTANCE                  200
#define SEA_LEVEL                               1013.25
#define VOLTAGE_SUPPLY                          5.1
#define BATTERY_MIN_VOLTAGE                 3.0
#define BATTERY_MAX_VOLTAGE                 4.2
#define A3144_TIMEOUT                           10 //time between read
#define WHEEL_DIAMETER                          1.7 //cm
#define OVERLOAD_PROTECTION_TIME            10
#define ENGINE_ACCELERATION_TIME            10
const float BAT_VOLTAGE_RANGE = BATTERY_MAX_VOLTAGE - BATTERY_MIN_VOLTAGE;
#define JSON_DATA_OUT_SIZE                  250
#define JSON_DATA_DETERMINATOR              '$'
#define JSON_DATA_TERMINATOR                '#'
#define SEND_DATA_TIME                          1000;
#define JSON_DATA_IN_SIZE                   50
//Pins
#define PIN_DHT                             2
#define PIN_US025_TRIG                          3
#define PIN_US025_ECHO                          4
#define PIN_ENG_1_1                         5 //right
#define PIN_ENG_1_2                         6 //right
#define PIN_ENG_2_1                         7 //left
#define PIN_ENG_2_2                         8 //left
#define PIN_A3144                               9
#define PIN_ENA                             10 //pwm
#define PIN_ENB                             11 //pwm
#define PIN_LED                             13
#define PIN_V_BAT                               14
#define PIN_V_SUPPLY                        15
```

```
//enums types
enum MoveSide
{
      FORWARD,//0
      BACK, //1
      LEFT, //2
      RIGHT,     //3
      STOP  //4
};
enum EngineType
{
      LEFT_ENG, //0
      RIGHT_ENG //1
};
//vars
DHT dht          (PIN_DHT, DHT11);
NewPing          sonar(PIN_US025_TRIG, PIN_US025_ECHO, MAX_US025_DISTANCE);
MPU6050          mpu(Wire);
Adafruit_BMP280 bmp;
StaticJsonDocument<JSON_DATA_OUT_SIZE> data_out;
StaticJsonDocument<JSON_DATA_IN_SIZE> data_in;
unsigned long timer_fast_sensor_update;
unsigned long timer_slow_sensor_update;
unsigned long timer_sensor_serial_print;
unsigned long timer_a3144_rps;
unsigned long timer_a3144_last_activation;
unsigned long timer_overload_protection = 0;
unsigned long timer_engine_acceleration = 0;
unsigned long timer_send_data;
volatile unsigned int turnovers = 0;
MoveSide current_move_side = STOP;
bool overload_protection_last_voltage_low = false;
bool engine_is_acceleration = false;
//data vars
int dht_huminidy = 0;
float dht_temperature = 0;
unsigned int us_distance = 0;
float mpu_temperature = 0;
float mpu_acc[3] = {0, 0, 0};
float mpu_gyro[3] = {0, 0, 0};
float mpu_angle[3] = {0, 0, 0};
float bmp_temperature = 0;
float bmp_pressure = 0;
float bmp_altitude = 0;
float bat_voltage = 0;
int bat_percente = 0;
float supply_voltage = 0;
unsigned int rps = 0;
float speed = 0; // cm/s
float mileage = 0;// cm
void setup()
```

```
{
	pinMode(PIN_LED, OUTPUT);
	Blink();

	Serial.begin(SERIAL_SPEED);
	Serial.setTimeout(SERIAL_TIMEOUT);
	Wire.begin();
	EnginesInit();
	SensorsInit();
	SendDataInit();

	Blink();
	Serial.println("Init OK!");
};
void loop()
{
	SensorsUpdate();
	//TestEngines();
	OverloadProtection();
	SendData();
	ReceiveData();
}
void ReceiveData()
{
	if(Serial.available())
	{
		if (Serial.read() == JSON_DATA_DETERMINATOR)
		{
			Serial.flush();//wait for end transmition
			String bufer =
Serial.readStringUntil(JSON_DATA_TERMINATOR);
			//Serial.println();
			//Serial.println(bufer);
			//Serial.println();
			DeserializationError err = deserializeJson(data_in,
bufer);
			Move((MoveSide)data_in["control"]);

			if (err == DeserializationError::Ok)
			{
				//Serial.println("JSON INPUT\n");
				//serializeJson(data_in, Serial);
				//Serial.println();
				//Send_GET_Data();
			}
			else
			{
				Serial.println("DeserializationError Code = " +
String(err.c_str()));
			}
		}
```

```
    }
}
void SendDataInit()
{
     timer_send_data = millis() + SEND_DATA_TIME;
}

void SendData()
{
     if (millis() > timer_send_data && SEND_DATA)
     {
          timer_send_data = millis() + SEND_DATA_TIME;

          FillOutJson();

          Serial.println();
          Serial.print(JSON_DATA_DETERMINATOR);
          serializeJson(data_out, Serial);
          Serial.print(JSON_DATA_TERMINATOR);
          Serial.println();
     }
}

void FillOutJson()
{
     data_out["tmp"] = dht_temperature;
     data_out["hum"] = dht_huminidy;
     data_out["u_dst"] = us_distance;
     data_out["spd"] = speed;
     data_out["mil"] = mileage;


     data_out["acc"][0] = mpu_acc[0];
     data_out["acc"][1] = mpu_acc[1];
     data_out["acc"][2] = mpu_acc[2];

     data_out["gyr"][0] = mpu_gyro[0];
     data_out["gyr"][1] = mpu_gyro[1];
     data_out["gyr"][2] = mpu_gyro[2];

     data_out["ang"][0] = mpu_angle[0];
     data_out["ang"][1] = mpu_angle[1];
     data_out["ang"][2] = mpu_angle[2];

     data_out["prs"] = bmp_pressure;
     data_out["alt"] = bmp_altitude;
     data_out["bat"] = RoundFloatValue(bat_voltage);
     data_out["bpc"] = bat_percente;
}

void ReadBattery()
```

```
{
      bat_voltage = (float)(analogRead(PIN_V_BAT) * VOLTAGE_SUPPLY) / 1024;
      supply_voltage = (float)(analogRead(PIN_V_SUPPLY) * VOLTAGE_SUPPLY) /
1024;

      if (bat_voltage <= BATTERY_MIN_VOLTAGE)
      {
            bat_percente = 0;
      }
      else if (bat_voltage >= BATTERY_MAX_VOLTAGE)
      {
            bat_percente = 100;
      }
      else
      {
            //bat_percente = (float)(bat_voltage - BATTERY_MIN_VOLTAGE) /
(float)BAT_VOLTAGE_RANGE; //* 100;
            bat_percente = (float)(bat_voltage - BATTERY_MIN_VOLTAGE) /
BAT_VOLTAGE_RANGE * 100;
      }
}
void Interrupt_A3144()
{
      if (millis() > timer_a3144_last_activation + A3144_TIMEOUT)
      {
            turnovers++;
      }
}
void Read_A3144()
{
      if (millis() > timer_a3144_rps)
      {
            timer_a3144_rps = millis() + 1000;

            rps = turnovers;
            speed = rps * WHEEL_DIAMETER;
            mileage += speed;

            turnovers = 0;
      }
}
void TestEngines()
{
      if (Serial.available() > 0)
      {
            char c = Serial.read();
            {
                  if (c != '\r' && c != '\n')
                  {
                        MoveSide side = (MoveSide)(c - '0');
                        Move(side);
```

```
                }
            }
        }
}
void Blink()
{
    Blink(10, 100);
}
void Blink(int count, int _delay)
{
    bool led = true;
    for (int i = 0; i < count; i++)
    {
        digitalWrite(PIN_LED, led);
        delay(_delay);
        led = !led;
    }
}
void EnginesInit()
{
    pinMode(PIN_ENG_1_1, OUTPUT);
    pinMode(PIN_ENG_1_2, OUTPUT);
    pinMode(PIN_ENG_2_1, OUTPUT);
    pinMode(PIN_ENG_2_2, OUTPUT);
    pinMode(PIN_ENA, OUTPUT);
    pinMode(PIN_ENB, OUTPUT);
    digitalWrite(PIN_ENG_1_1, LOW);
    digitalWrite(PIN_ENG_1_2, LOW);
    digitalWrite(PIN_ENG_2_1, LOW);
    digitalWrite(PIN_ENG_2_2, LOW);
    analogWrite(PIN_ENA, 255);
    analogWrite(PIN_ENB, 255);
}

void Move(MoveSide side)
{
    if (side != current_move_side)
    {
        turnovers = 0;
    }

    if (side != STOP)
    {
        EngineMove(LEFT_ENG, STOP);
        EngineMove(RIGHT_ENG, STOP);
        delay(30);

        if (side != current_move_side)
        {
            engine_is_acceleration = true;
```

```
                timer_engine_acceleration = millis() +
ENGINE_ACCELERATION_TIME;
            }
      }
      switch(side)
      {
            case FORWARD:
                  EngineMove(LEFT_ENG, FORWARD);
                  EngineMove(RIGHT_ENG, FORWARD);
                  break;
            case BACK:
                  EngineMove(LEFT_ENG, BACK);
                  EngineMove(RIGHT_ENG, BACK);
                  break;
            case LEFT:
                  EngineMove(LEFT_ENG, BACK);
                  EngineMove(RIGHT_ENG, FORWARD);
                  break;
            case RIGHT:
                  EngineMove(LEFT_ENG, FORWARD);
                  EngineMove(RIGHT_ENG, BACK);
                  break;
            case STOP:
                  EngineMove(LEFT_ENG, STOP);
                  EngineMove(RIGHT_ENG, STOP);
                  break;
            default:
                  Serial.println("ERORR: Unknow move side, Move()");
                  break;
      }
      current_move_side = side;
}
void EngineMove(EngineType eng, MoveSide side)
{
      int pin_1 = -1;
      int pin_2 = -1;

      if(eng == RIGHT_ENG)
      {
            pin_1 = PIN_ENG_1_1;
            pin_2 = PIN_ENG_1_2;
      }
      else if(eng == LEFT_ENG)
      {
            pin_1 = PIN_ENG_2_1;
            pin_2 = PIN_ENG_2_2;
      }
      else
      {
            Serial.println("ERORR: Unknow engine type, EngineMove()");
            return;
```

```
	}
	switch(side)
	{
		case FORWARD:
			digitalWrite(pin_2, LOW);
			digitalWrite(pin_1, HIGH);
			break;
		case BACK:
			digitalWrite(pin_1, LOW);
			digitalWrite(pin_2, HIGH);
			break;
		case STOP:
			digitalWrite(pin_1, LOW);
			digitalWrite(pin_2, LOW);
			break;
		default:
			Serial.println("ERORR: Unknow move side, EngineMove()");
			break;
	}
}
void SensorsInit()
{
	//DHT
	dht.begin();
	//MPU
	mpu.begin();
	Serial.println("Calibration MPU6050...");
	mpu.calcOffsets(true, true);
	delay(1000);
	//BMP
	bmp.begin(BMP280_ADDRESS_ALT, BMP280_CHIPID);
	bmp.setSampling(Adafruit_BMP280::MODE_NORMAL,     /* Operating Mode.
*/
			Adafruit_BMP280::SAMPLING_X2,     /* Temp.
oversampling */
			Adafruit_BMP280::SAMPLING_X16,    /* Pressure
oversampling */
			Adafruit_BMP280::FILTER_X16,      /* Filtering.
*/
			Adafruit_BMP280::STANDBY_MS_500); /* Standby
time. */
	attachPCINT(digitalPinToPCINT(PIN_A3144), Interrupt_A3144, FALLING);

	timer_fast_sensor_update = millis() + FAST_SENSOR_UPDATE_TIME;
	timer_slow_sensor_update = millis() + SLOW_SENSOR_UPDATE_TIME;
	timer_sensor_serial_print = millis() + SENSOR_SERIAL_PRINT_TIME;
	timer_a3144_rps = millis() + 1000;
	timer_a3144_last_activation = 0;
}
void SensorsUpdate()
{
```

```
	//FAST SENSORS
	if (millis() > timer_fast_sensor_update)
	{
		timer_fast_sensor_update = millis() + FAST_SENSOR_UPDATE_TIME;

		//BATTERY POWER
		ReadBattery();
		//SONAR
		us_distance = sonar.ping_cm();
		//MPU6050
		mpu.update();
		mpu_temperature = mpu.getTemp();
		mpu_acc[0] = RoundFloatValue(mpu.getAccX());
		mpu_acc[1] = RoundFloatValue(mpu.getAccY());
		mpu_acc[2] = RoundFloatValue(mpu.getAccZ());
		mpu_gyro[0] = RoundFloatValue(mpu.getGyroX());
		mpu_gyro[1] = RoundFloatValue(mpu.getGyroY());
		mpu_gyro[2] = RoundFloatValue(mpu.getGyroZ());
		mpu_angle[0] = RoundFloatValue(mpu.getAngleX());
		mpu_angle[1] = RoundFloatValue(mpu.getAngleY());
		mpu_angle[2] = RoundFloatValue(mpu.getAngleZ());
		//BMP280
		bmp_temperature = bmp.readTemperature();
		bmp_pressure = bmp.readPressure();
		bmp_altitude = RoundFloatValue(bmp.readAltitude(SEA_LEVEL));

		Read_A3144();
	}
	//SLOW SENSORS
	if (millis() > timer_slow_sensor_update)
	{
		timer_slow_sensor_update = millis() + SLOW_SENSOR_UPDATE_TIME;

		//DHT11 - 25 ms read
		dht_huminidy = (int)dht.readHumidity();
		dht_temperature = dht.readTemperature();
	}
	//SERIAL PRINT
	if (millis() > timer_sensor_serial_print && SENSORS_SERIAL)
	{
		timer_sensor_serial_print = millis() + SENSOR_SERIAL_PRINT_TIME;

		SensorsSerialPrint();
	}
}
void SensorsSerialPrint()
{
	Serial.println("Hum = " + String(dht_huminidy) +  "\tTemp = " +
String(dht_temperature));
	Serial.println("Distance = " + String(us_distance));
	Serial.println("MPU temperature = " + String(mpu_temperature));
```

```
      Serial.println("Acc = X: " + String(mpu_acc[0]) + " Y:" +
String(mpu_acc[1]) + " Z:" + String(mpu_acc[2]));
      Serial.println("Gyro = X: " + String(mpu_gyro[0]) + " Y:" +
String(mpu_gyro[1]) + " Z:" + String(mpu_gyro[2]));
      Serial.println("Angle = X: " + String(mpu_angle[0]) + " Y:" +
String(mpu_angle[1]) + " Z:" + String(mpu_angle[2]));
      Serial.println("BMP: temp = " + String(bmp_temperature) +
"\tPressure = " + String(bmp_pressure) +"\tAltitude = " +
String(bmp_altitude));
      Serial.println("BAT voltage = " + String(bat_voltage) + "\tBAT = " +
String(bat_percente) + '%' + "\tVSupply = " + String(supply_voltage));
      Serial.println("RPS = " + String(rps) + "\tSPD = " + String(speed) +
"\tMLG = " + String(mileage));
      Serial.println();
}
void OverloadProtection()
{
      if (engine_is_acceleration)
      {
            if (millis() > timer_engine_acceleration)
            {
                  engine_is_acceleration = false;
            }
      }
      else
      {
            if (millis() > timer_overload_protection)
            {
                  timer_overload_protection = millis() +
OVERLOAD_PROTECTION_TIME;

                  supply_voltage = (float)(analogRead(PIN_V_SUPPLY) *
VOLTAGE_SUPPLY) / 1024;
                  if (supply_voltage <= VOLTAGE_SUPPLY - 0.03)
                  {
                        Move(STOP);
                  }
            }
      }
}
float RoundFloatValue(float value)
{
      return ((float)(int)(value * 100)) / 100;
}
```

Національний університет «Полтавська політехніка імені Юрія Кондратюка»
Навчально-науковий інститут інформаційних технологій і робототехніки
Кафедра автоматики, електроніки та телекомунікацій

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

НА ТЕМУ **РОЗРОБКА САМОХІДНОЇ, ТЕЛЕМЕТРИЧНОЇ, РАДІОКЕРОВАНОЇ СИСТЕМИ**

Виконав: студент групи 401-ТТ Карпук В.Ю.
Керівник кваліфікаційної роботи: Обіход Я.Я.

Полтава 2021

ТЕМА: РОЗРОБКА САМОХІДНОЇ, ТЕЛЕМЕТРИЧНОЇ, РАДІОКЕРОВАНОЇ СИСТЕМИ

Мета: розробка проекту самохідної, телеметричної, радіокерованої системи та додатку керування для неї.

Об’єкт дослідження: дистанційно керована платформа із можливостями телеметрії.

Предмет дослідження: основні принципи побудови дистанційно керованих телеметричних систем.

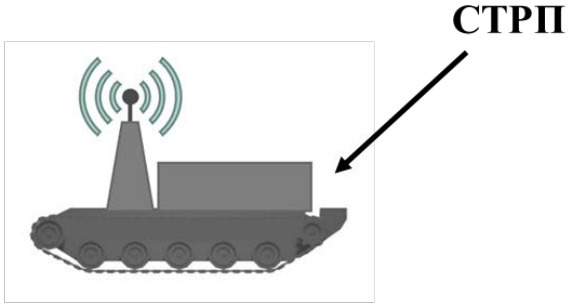
Задачі даної кваліфікаційної роботи:

- Аналіз сучасних дистанційних, телеметричних систем.
- Створення загальних принципів побудови дистанційно керованих систем.
- Вибір обладнання для дистанційної системи.
- Розробка проекту самохідної, телеметричної, радіокерованої системи.
- Розробка додатку керування для радіокерованої системи.

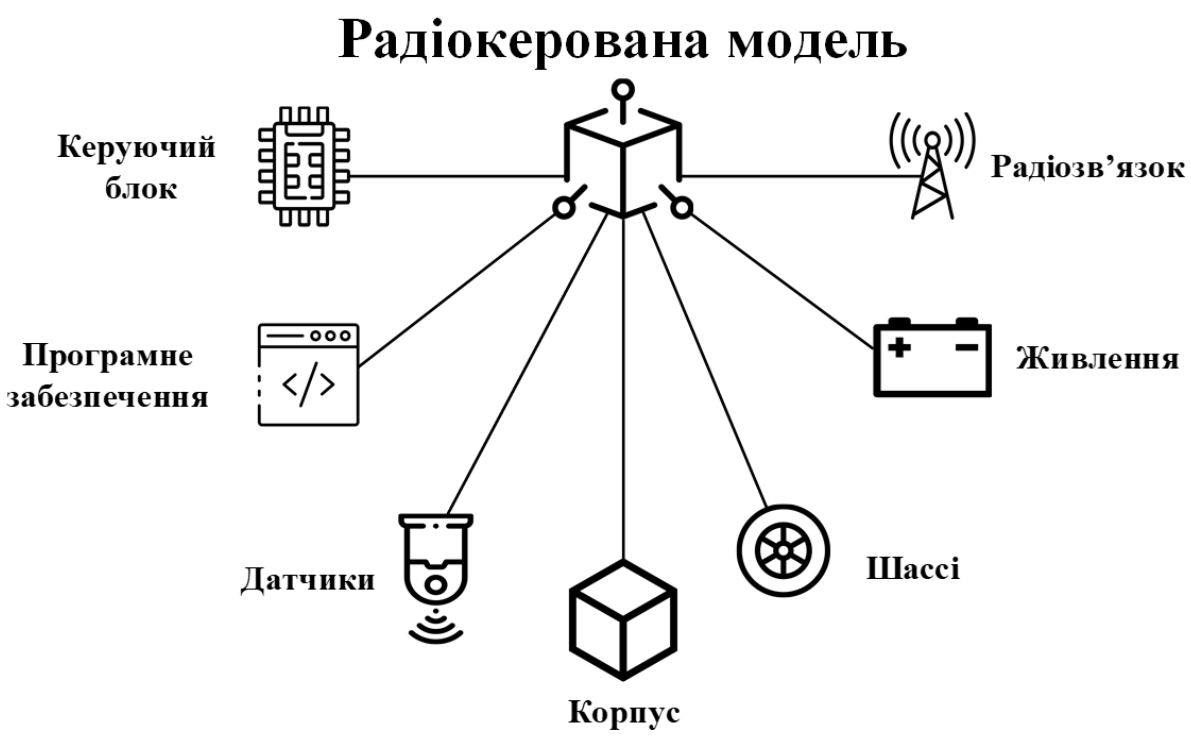
ОСНОВНА ТЕОРІЯ

- **Телеметричне керування** – дистанційне керування із можливістю відображення показників датчиків та відеокамер.

- **СТРП** – самохідна телеметрична радіокерована платформа.



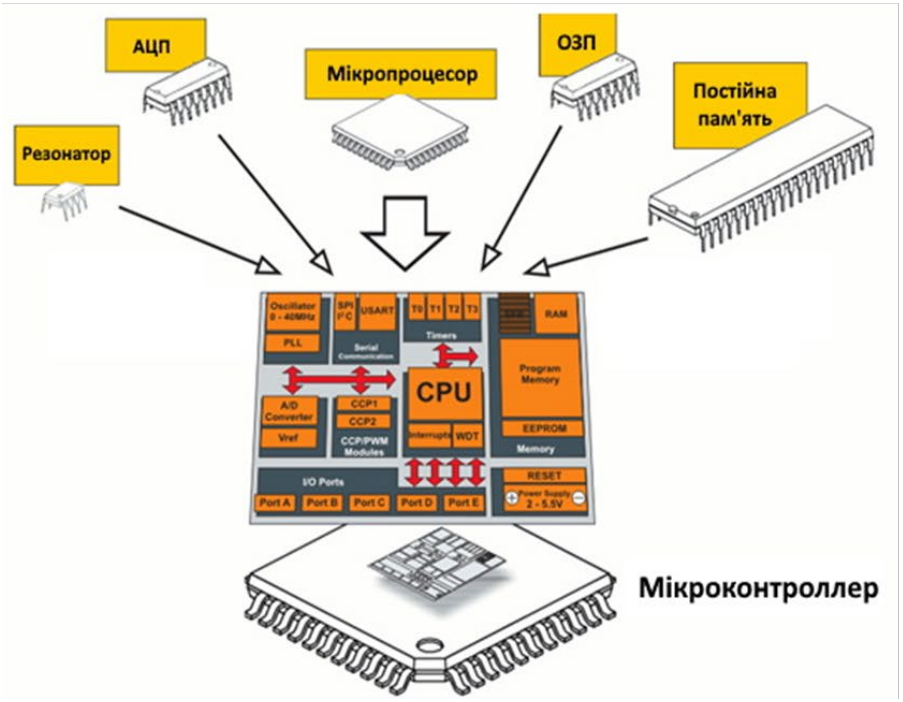
ПРИНЦИПИ ПРОЕКТУВАННЯ РАДІОКЕРОВАНИХ МОДЕЛЕЙ



Шасі для побудови СТРП

БЛОК КЕРУВАННЯ СТРП

Будова сучасного мікроконтролеру



Мікроконтролери у складі СТРП – їх призначення

ESP32-S

– Wi-Fi зв’язок та відеокамера

MEGA 328P

– датчики та керування

ТЕЛЕМЕТРИЧНІ МОЖЛИВОСТІ СТРП

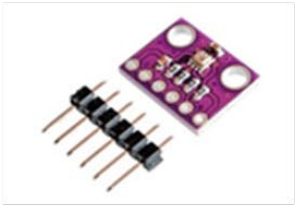
Дані, що збираються:

- температура,
- вологість,
- відстань до перешкоди,
- швидкість,
- пробіг,
- кут нахилу,
- тиск,
- висота над рівнем моря,
- напруга батареї,
- рівень сигналу.

US-025
Ультразвуковий далекомір

DHT-11
Термометр та гігрометр

OV2640
Модуль камери



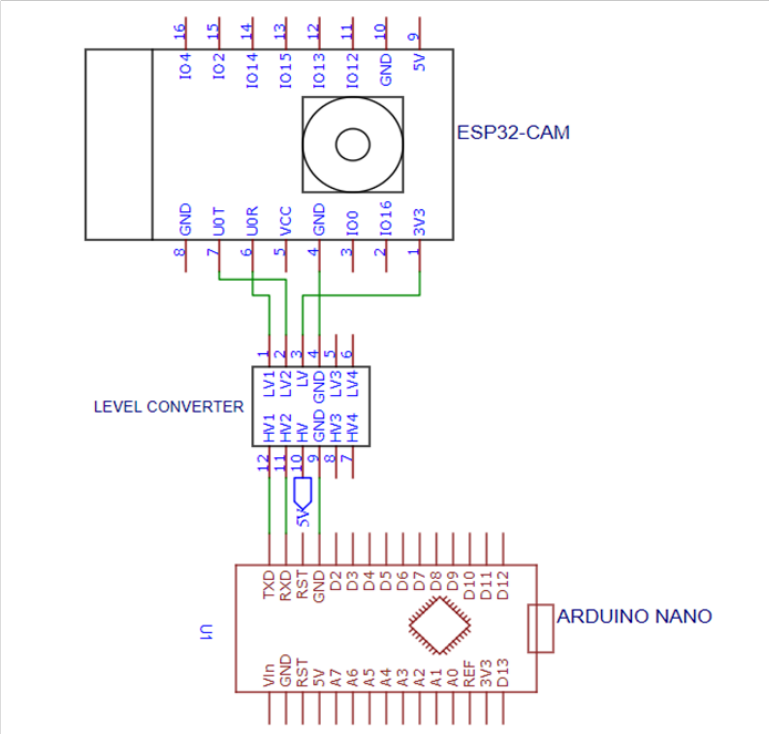
BMP-280
Цифровий барометр

MPU-6050
Акселерометр та гіроскоп

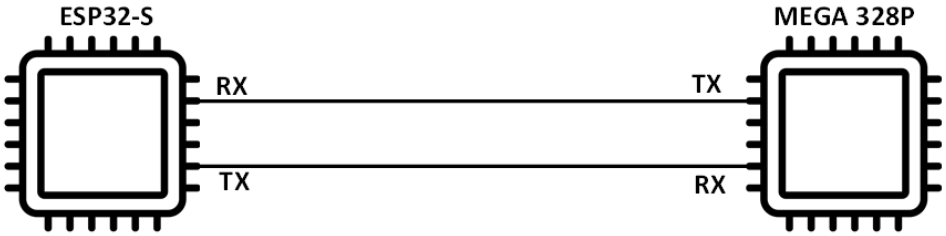
A3144
Датчик Холлу

РЕАЛІЗАЦІЯ ЗВ'ЯЗКУ ПО ПРОТОКОЛУ UART

Електрична принципова схема



Логічна схема



Приклад JSON коду

{"sensor": 53, "control": 2}

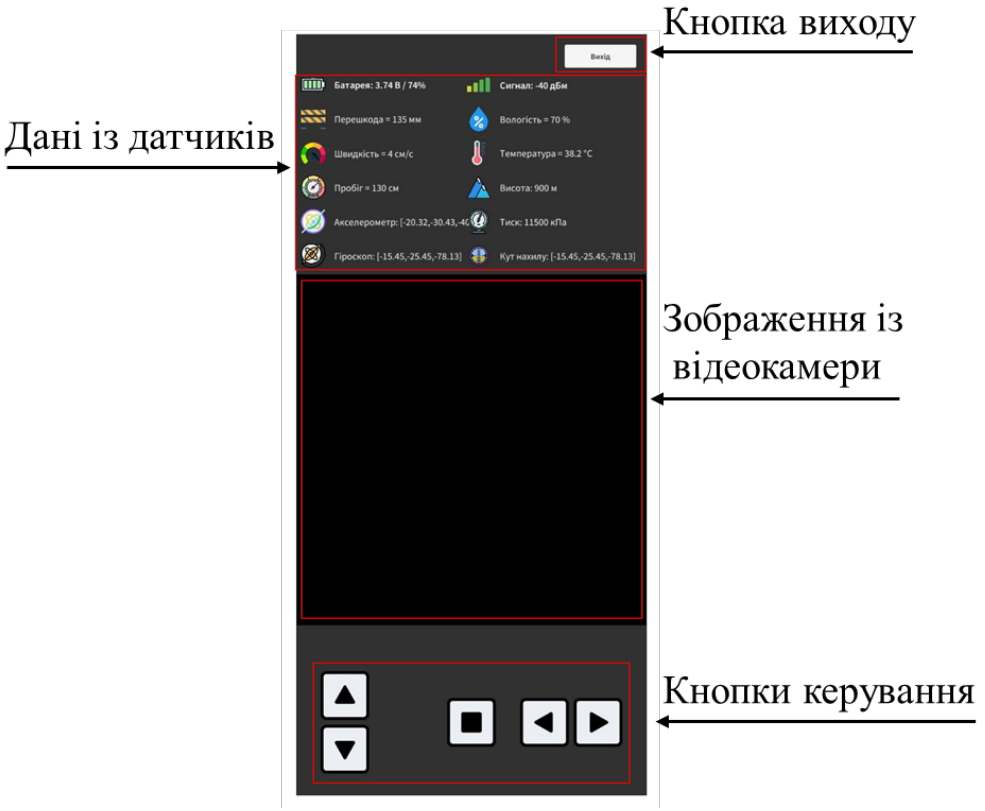
Величина із датчику = 53

Сигнал керування = поворот ліворуч

РОЗРОБКА ДОДАТКУ КЕРУВАННЯ

Інструменти для розроблення:

- IDE MS Visual Studio Community.
- Unity – програмне середовище для розробки інтерактивного контенту.
- Мова програмування C#.



РЕАЛІЗАЦІЯ ЗВ’ЯЗКУ МІЖ СТРП ТА ДОДАТКОМ КЕРУВАННЯ

Умови передачі:

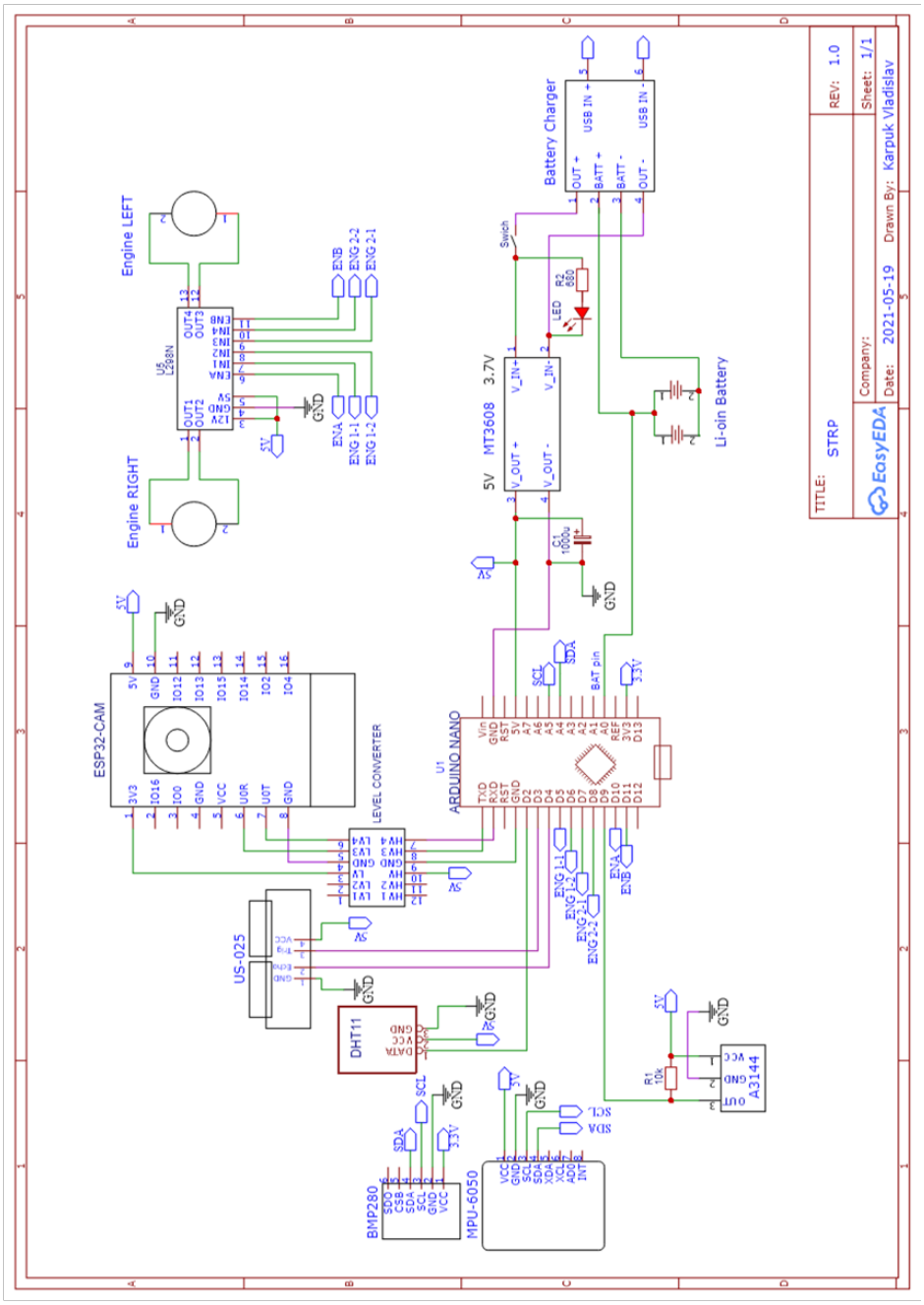
- Якщо наявний сигнал керування він буде негайно переданий.
- Опитування із деякою періодичністю СТРП на наявність нових даних від датчиків

ЗВ’ЯЗОК ІЗ СТОРОННІМИ СЕРВІСАМИ

Віддалений сервіс через HTTP запити має можливість:

- Отримувати дані від датчиків.
- Відтворювати відео від камери СТРП.
- Відправляти команди керування СТРП.

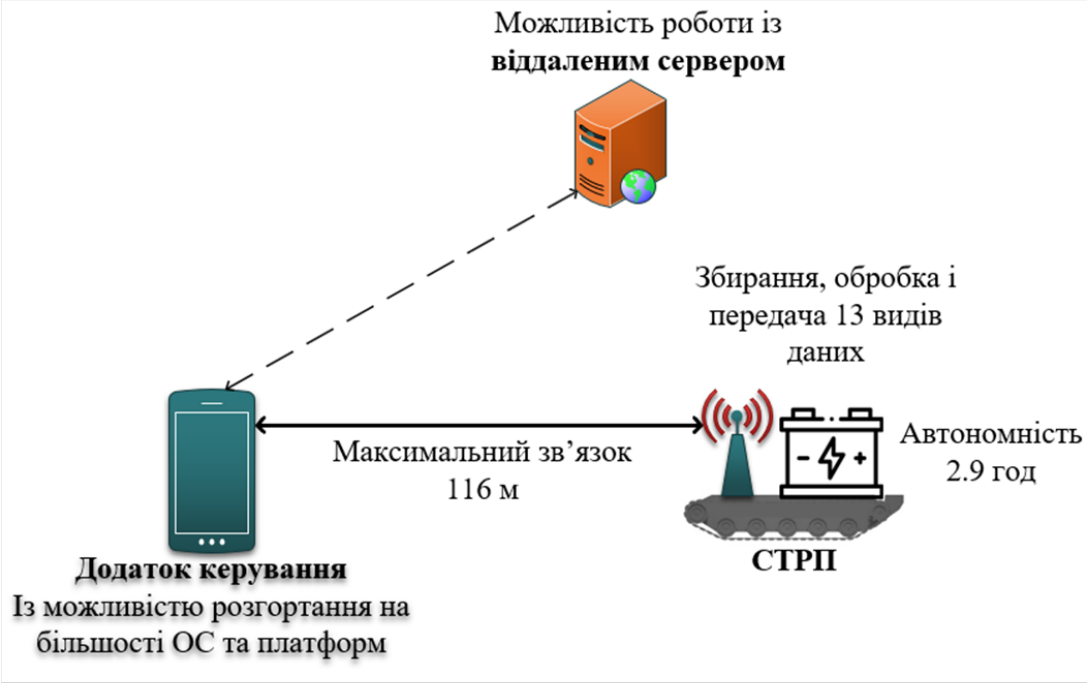
ЗАГАЛЬНА ЕЛЕКТРИЧНА ПРИНЦИПОВА СХЕМА СТРП



ВИСНОВКИ

У ході виконання кваліфікаційної роботи було виконано наступні завдання:

- Проведено аналіз сучасних дистанційно керованих систем, на основі чого виведено принципи побудови подібних систем.
- Вибрано обладнання для побудови дистанційної системи.
- Розроблено проект самохідної радіокерованої системи та додатку керування для неї.



Дякую за увагу!