Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій

**МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА**

на тему:

# «Портативна система визначення наявності джерела радіочастотного випромінювання»

Виконав: студент 2 курсу, групи 1АКІТ-22м,
спеціальність 151– «Автоматизація та комп'ютерно-інтегровані технології».
_____ Юрій КУЗІН

Керівник к.т.н., доцент кафедри АІІТ
_____ Володимир ГАРМАШ
«14» грудня 2023 р.

Опонент: к.т.н. доцент кафедри КН
Руслан Белзецький
«4» грудня 2023 р.

**Допущено до захисту**
Завідувач кафедри АІІТ

_____ д.т.н., проф. Олег БІСІКАЛО
«14» 12 2023 р.

Вінниця ВНТУ – 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій
Рівень вищої освіти 2-й (магістерський)
Галузь знань – 15 Автоматизація та приладобудування
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
Освітньо-професійна програма - Інтелектуальні комп'ютерні системи

**ЗАТВЕРДЖУЮ**
Завідувач кафедри АІІТ
Олег БІСІКАЛО
(прізвище та ініціали)
«20» 09 2023 року

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

Кузін Юрій Володимирович
(прізвище, ім'я, по батькові)

1. Тема роботи: «Портативна система визначення наявності джерела радіочастотного випромінювання»
керівник роботи:
Гармаш Володимир Володимирович, к.т.н., доцент кафедри АІІТ
( прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від «18» 09 2023 року № 247

2. Строк подання студентом роботи 20 листопада 2023

3. Вихідні дані до роботи: стек технологій для розробки продукту; технічна документація та специфікація елементної бази; частотний діапазон 2.4-6GHz; відстань елементів системи 10 км.

4. Зміст текстової частини: Вступ; Аналіз сучасного стану питання та необхідності створення системи визначення місцеперебування джерела радіочастотного випромінювання; Розробка структури системи визначення джерела радіовипромінювання; Розробка програмного забезпечення системи; Аналіз результатів; Економічний розділ. Висновки. Список використаних джерел.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень): Схема електрична принципова пристрою турелі; Схема електрична принципова пристрою приймача; Креслення корпусу пристрою турелі; Креслення поворотної частини турелі; Креслення статичної частини турелі; Алгоритм функціонування системи.

6. Консультанти розділів роботи

| Розділ змістової частини роботи | Ім`я ПРІЗВИЩЕ, та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| Розділи 1, 2, 3, 4 | доцент кафедри АІІТ, доцент, Володимир ГАРМАШ | 21.09.2023 | 04.12.2023 |
| Економічний розділ | професор кафедри ЕПВМ, Професор, к.е.н Володимир КОЗЛОВСЬКИЙ | 21.09.2023 | 06.12.2023 |

7. Дата видачі завдання «26» 09 2023 року

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів магістерської кваліфікаційної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Аналіз сучасного стану питання | 13.10.23 | вик |
| 2 | Вибір методів імплементації та елементної бази | 13.10.23 | вик |
| 3 | Розробка схеми електричної принципової | 01.12.23 | вик |
| 4 | Розробка програмного забезпечення | 15.11.23 | вик |
| 5 | Підготовка економічного розділу | 01.12.23 | вик |
| 6 | Оформлення пояснювальної записки, графічного матеріалу і презентації | 01.12.23 | вик |
| 9 | Графічні матеріали | 04.12.23 | вик |
| 10 | Захист МКР | з 11.12.23 по 29.12.23 | вик |

Студент _____ (підпис) Юрій КУЗІН (ім'я прізвище)

Керівник роботи _____ (підпис) Володимир ГАРМАШ (ім'я прізвище)

# АНОТАЦІЯ

УДК 004.9

Кузін Ю.В. Портативна система визначення наявності джерела радіочастотного випромінювання. Магістерська кваліфікаційна робота зі спеціальності 151 — Автоматизація та комп’ютерно-інтегровані технології, освітня програма — Інтелектуальні комп’ютерні системи. Вінниця ВНТУ, 2023. 137с.

На укр.мові.Бібліогр.:67;рис.:46; табл.:9

Метою роботи є створення портативної системи визначення напрямку місцеперебування джерел радіочастотного випромінювання з використанням мікроконтролерних технологій, технологій дальнього зв’язку та систем позиціювання.

У загальній частині розглянуто методи визначення джерел радіочастотного випромінювання, технології розробки таких систем, обґрунтована доцільність розробки такої системи.

Ключові слова: портативна система, радіочастота, джерело випромінювання, вимірювання.

# ANNOTATION

Kuzin Y.V. Portable system of radiofrequency source availability. Master's thesis on speciality 151 — Automation and computer-integrated technologies, Educational program — Itelegent computer systems. Vinnytsia: VNTU, 2023. 137p.

In the Ukrainian language. Bibliography.:67;fig.:46;tabl.:9

The main target of this work it is developing of portable system for direction identification of radiofrequency source. The system will be microcontroller based with using long range conection and global positioning technologies.

The main part of this work describes methodes of radiofrequency sources identification, technologies that is used for such systems and expidiency of such system development.

Key words: portable system, radiofrequency, RF radiation sources, measurement

# ЗМІСТ

# ВСТУП

*Актуальність роботи.* Радіочастотне випромінювання є невід'ємною частиною сучасного світу, яке використовується для забезпечення різних видів зв'язку, навігації, радіолокації, дистанційного керування пристроями та інших цілей [1]. Однак, радіочастотне випромінювання також може становити потенційну загрозу для безпеки людини, оскільки може бути використане для проведення шпигунської, терористичної діяльності, тобто опосередкована небезпека, де засобом впливу є пристрої та речовини, що доставляють за допомогою радіо-керованих пристроїв. Наприклад, безпілотні пристрої можуть порушувати повітряний простір і при цьому нести шкідливі речовини чи засоби фізичного ураження [2]. Також існує пряма небезпека, що несе з собою радіо випромінювання, це наприклад несанкціоноване використання джерел потужного радіовипромінювання, що може створювати перешкоди для санкціонованого використання радіочастотного спектру і також спричиняти шкоду здоров'ю людей.

Саме тому актуальною є задача виявлення та локалізації джерел радіочастотного випромінювання [3, 4], що дозволить ідентифікувати їх джерело, характеристики та напрямок дії, а також прийняти заходи щодо їх нейтралізації або усунення. Для цього необхідна портативна система визначення наявності джерела радіочастотного випромінювання, яка б могла працювати в реальному часі, мати високу точність та чутливість, бути компактною та легкою, а також мало залежати від умов навколишнього середовища і залишати мінімальний радіо слід.

*Метою* магістерської кваліфікаційної роботи є програмно-апаратна розробка такої портативної системи, яка може ефективно впоратися з ідентифікацію джерел радіочастотного випромінювання, що порушують повітряний простір та чистоту радіочастотного діапазону. Шляхом

використання як новітніх так і перевірених часом технологій у галузі сенсорів та обробки сигналів.

*Об'єктом досліджень* є процес вимірювання потужності радіочастот для визначення напрямку перебування джерела радіочастотного випромінювання з метою покращення ідентифікації активних об'єктів випромінювання, що порушують правила користування повітряним простором та радіочастотним діапазоном.

*Предметом досліджень* є методи та засоби реалізації системи визначення напрямку перебування джерела радіочастотного випромінювання.

*Задачі досліджень* магістерської кваліфікаційної роботи:

1. Вивчення наявних імплементацій пристроїв визначення місцеположення, з можливістю передавання інформації з допомогою LoRa, та аналіз їх недоліків.

2. Дослідження та вибір наявних технологій та апаратних пристроїв, що відповідають висунутим вимогам таких як використання GPS [5, 6], портативність буде забезпечена за допомогою технології LoRa [7, 8] передачі інформації з координатами станції на відстань та виміряними потужностями, від кінцевого пристрою до оператора чи до станції контролю, в зашифрованому вигляді.

3. Техніко-економічне та науково-технічне обґрунтування конфігурації системи.

4. Розробка технічного завдання на науково-дослідну роботу.

5. Проектування апаратної частини пристрою.

6. Проектування програмної частини пристрою.

*Новизна* полягає в креативності ідеї створення портативної системи для визначення місцеперебування джерел радіовипромінювання, з можливістю розширення системи та створення віртуальної мережі пристроїв, планується розробка алгоритмів оптимізації управління модулями та режимами роботи системи, імплементація заходів захисту, імплементація протоколів спілкування

елементів системи, збільшення радіусу дії та зменшення ціни готового пристрою завдяки вибору оптимальних рішень та компонентів для їх реалізації.

*Практичне значення,* що планується отримати в ході роботи, полягає в наступному: розробка портативної системи визначення наявності джерела радіочастотного випромінювання, яка може працювати в реальному часі, мати високу точність та чутливість, бути компактною та легкою, а також мало залежати від умов навколишнього середовища і залишати мінімальний радіо слід. Розробка алгоритмів оптимізації управління модулями та режимами роботи пристрою, імплементація заходів захисту, збільшення радіусу дії та зменшення ціни готового пристрою. Проектування апаратної та програмної частин пристрою, що використовують сучасні досягнення у галузях радіотехніки, електроніки, антенної техніки, обробки сигналів, математичного моделювання та інших. Проведення експериментальних досліджень пристрою та аналіз його характеристик та ефективності.

*Практичною цінністю* даної роботи є розроблена система її програмне та алгоритмічне забезпечення, яке може бути застосоване для забезпечення безпеки повітряного простору об’єктів під охороною.

# 1 АНАЛІЗ СУЧАСНОГО СТАНУ ПИТАННЯ ТА НЕОБХІДНОСТІ СТВОРЕННЯ СИСТЕМИ ВИЗНАЧЕННЯ МІСЦЕПЕРЕБУВАННЯ ДЖЕРЕЛА РАДІОЧАСТОТНОГО ВИПРОМІНЮВАННЯ

В сучасних реаліях, коли різні БПЛА набули широкого вжитку і їх різноманітність все збільшується, збільшується і кількість загроз які вони несуть з собою, це як несанкціоноване втручання в повітряний простір об’єктів, що охороняються так і можливість нести небезпечні речовини і завдавати шкоди, також зі зростанням різноманітності БПЛА, зростає кількість пристроїв глушіння радіочастотного спектру та збільшується кількість випадків їх довільного використання, що часто спричиняє завади для санкціонованої діяльності.

Для виявлення джерела радіовипромінювання існують такі методи:

- оптичний — складність такої системи в необхідності високих обчислювальних потужностей, та навчання системи. Така система дуже залежить від погодних умов її використання не можливе у ночі. І розрізнити людину з вогнепальною рушницею чи джеммер рушницею на відстані майже не можливо, а виявлення невеличких за розміром БПЛА дуже лімітує відстань виявлення [9];

- тепловий — зазвичай цей спосіб є логічним продовженням оптичного, з можливістю використання за відсутності денного світла, значно зменшує вплив погодних умов, але все ж таки залишається дуже чутливою до них. Проблема виявлення і розпізнавання випромінювачів залишається актуальною, як і у випадку з оптичною. А маленький квадрокоптер в повітрі взагалі має невеличкий тепловий слід;

- акустичний — метод може бути використаний лише для виявлення БПЛА [10], але використання його в регіонах з великим шумовим забрудненням, поряд з

дорогами чи місцями де є багато вибухів, унеможливлює навіть ідентифікацію БПЛА, разом оптичним і тепловим цей метод є більше методом опосередкованого пошуку джерела радіочастотного випромінювання за певними ознаками [11];

- радіолокаційний — саме цей метод є найбільш розповсюдженішим для виявлення джерел радіовипромінювання, а правильніше сказати методи, оскільки даний термін можна як мінімум розподілити два основних типи це методи активної і пасивної радіолокації.

Активний метод радіолокації прикладом якого є звичайний радар, є сам джерелом радіовипромінювання і не є надійним способом виявлення невеличких БПЛА, оскільки їх площа дуже маленька і відбитого сигналу зазвичай недостатньо, щоб виявити такий пристрій у повітрі на більш-менш прийнятній відстані, а у випадку наземних станцій глушіння він взагалі не є ефективний.

Пасивний метод радіолокації, до цього пункту можна віднести всі методи які базуються на прийманні радіосигналів, це можуть бути складні методи радіопеленгу, які мало того що потребують відносно дорогого обладнання та обчислювальних потужностей, необхідний мінімум для цього це наприклад SDR (Software Defined Radio) такий як перевірений часом hackRF [12], чи досить свіжа розробка TinySA ULTRA, що базується на антенному аналізаторі nanoVNA [13]. Обидві платформи є з відкритою елементною базою та сирцями. Tiny SA ULTRA перевищує за деякими показниками hack RF та вже має екран для виводу параметрів, що вимірюються, але ціна таких платформ більша 100 американських доларів, і купити Tiny SA Ultra по ціні менше 130$ це вже досить хороша опція [14]. Існують деякі ідеї та дослідження, можливості ідентифікації дронів за допомогою SDR, ці методи мають свої переваги, наприклад як аналіз радіочастотного трафіку, дозволяє іноді отримати GPS координати комерційних дронів, але це потребує попереднього аналізу і

виокремлення протоколу, а так як ми згадали, що ціна SDR досить висока то і ціна таких методів значно виростає [15]. Аналіз частот, фазового зсуву та інших параметрів потребує, як мінімум наявності додаткового обчислювального блоку, що ще додатково збільшує ціну однієї станції. Цікава робота в сторону радіопеленгу була знайдена в КПІ і досить свіжа, що показує, що роботи в цьому напрямку в нашій країні ведуться [16]. І як робоча система, сама розробка відсутня, розробка лише на рівні структурної схеми, незважаючи, що модуль сповіщення вже реалізований, це як реалізація з кінця. Додатково до того такі методи ще й не завжди гарантують однозначне визначення напрямку сигналу і дуже піддаються впливу відбитих від перешкод сигналів. Ці ефекти можна нівелювати додаванням додаткових станцій, але це ще збільшить вимоги до системи в цілому, синхронізації та накладання отриманих результатів з кожної станції. В свою чергу специфічне відбивання сигналів від пропелерів, специфічний нахил дронів різних виробників, дозволяє дозволяє з помірною точністю їх розрізняти, цікаве дослідження в цю сторону проводились теж в але знову ж таки необхідна наявність SDR, а складні алгоритми потребують значних потужностей, а зважаючи на швидкий ріст індустрії дронів, і їх різноманітність, корисність такого підходу викликає питання, оскільки кожен новий дрон вимагає наповнення бази в лабораторних умовах [17]. Також до пасивних методів можна віднести методи визначення місцеперебування за RSSI хоча для прийому використовуються активні пристрої, і пристрої саме специфічно тієї ж частоти, що й дрон. Це відповідно накладає обмеження визначення місцеперебування дронів до дуже вузької лінійки моделей. Та і приймачі повинні бути розташовані на достатній відстані один від одного, що додає проблем в створені такої мережі, чим більша кількість пристроїв приймачів тим краща точність, при цьому використовуються складні математичні розрахунки [18].

Звісно, що існують робочі системи, що використовують як активні так і пасивні методи і додатково ще й містять системи радіоелектронної боротьби,

такі системи зазвичай коштують дуже дорого, як у виробництві так і в експлуатації, потребують великої кількості навченого персоналу, прикладом такої системи може бути, захоплена в Україні, ворожа «Барісаґлєбск-2» [19].

Альтернативою таким методам, може бути проста пасивна радіолокаційна система, що буде базуватись на демодулюючому логарифмічному підсилювачі, даний пристрій, призначений для точного перетворення отриманого радіочастотного сигналу в відповідне децибел-скальоване значення напруги [20]. Розширення такої системи додатковими станціями не потребує значних витрат, оскільки вся станція може контролюватись малопотужним мікроконтролером типу ATmega, PIC, STM, ESP ціна яких рідко перевищує 10$, а зазвичай найпростіших моделей наближається до 2$, і є тенденція до зниження [21]. Діапазон фіксації радіочастот такого пристрою буде залежати від характеристик логарифмічного перетворювача та характеристик антени, сама станція може бути також розширена шляхом збільшення кількості логарифмічних перетворювачів та відповідно підібраних антен.

## 1.1 Аналіз фреймворків та платформ мікроконтролерів для реалізації апаратного забезпечення

Звісно для реалізації можна використовувати асемблер чи мову призначену для конкретного мікроконтролера. Але в цьому є великі недоліки, такі як неможливість швидко змінити платформу на оновлену чи більш потужнішу, чи можливо зовсім інше сімейство мікроконтролерів. А також час затрачений на імплементацію роботи з різними датчиками та пристроями буде досить великий. Не зважаючи на те, що на асемблері можливо ідеально оптимізувати код, все ж таки доцільніше використовувати якусь з доступних платформ розробки, щоб забезпечити легке оновлення чи зміну платформи-мікроконтролера.

Розглянемо найбільш популярніші з доступних недорогих платформ

1. Arduino досить широка лінійка плат, що базується на сімействі мікроконтролерів Microchip ATmega (та деяких мікроконтролерів аналогів схожих по функціоналу) [22].

   Мова: рідна мова розробки C/C++ (Arduino IDE). Arduino IDE має широку спільноту користувачів, і велику кількість платформ що підтримуються.

2. ESP32 теж широка лінійка плат для розробки як виходить з назви базується на мікроконтролерах сімейства espressif ESP32 [23]. Досить дешева платформа не настільки як більшість Arduino, що має більшу частоту CPU порівняно з Arduino навіть в платформах з низькою потужністю споживання. Також має більший розмір пам’яті для програми і більший розмір оперативної пам’яті.

   Мова: рідна мова розробки LUA але також офіційно підтримується Arduino IDE. LUA не досить велика і окрім всього це інтерпретована

мова, але ці недоліки компенсуються офіційною підтримкою Arduino IDE, що повертає нас до C/C++.

3. STM32 плати розробника що базуються на STMicroelectronics STM32 [24]. Мікроконтролери теж мають зазвичай кращі параметри ніж Arduino і досить наближаються по характеристиках до ESP32 і приблизно той самий ціновий рівень [25].

   Мова: рідна мова C/C++ для своєї власної STM32CubeIDE [26] але існує проект підтримки Arduino IDE, єдина умова використання це потрібно програматором прошити arduino bootloader [27].

4. Raspberry Pi, а саме плати, що базуються на мікроконтролері власної розробки RP2040 потужний процесор 133 MHz, що має два ядра, і достатньо пам'яті [28].

   Мова: рідна мова розробки пітон, а точніше мікропітон, який також є інтерпретованою скриптовою мовою, оскільки це пітон то існує широкий вибір IDE. Найлегша з них є Thonny IDE [29].

Потрібно зауважити, що зазвичай імплементація інтерпретатора мікропітону існує для багатьох з вищенаведених платформ, але оскільки пітон інтерпретована мова і в умовах обмежених ресурсів і відсутності можливості прямого управління пам’яттю інколи можна отримувати неочікувані результати.

Виходячи з вищенаведеного, що найкращу портативність коду забезпечує Arduino IDE C/C++, більш широку спільноту розробників і готових модулів для роботи з датчиками, і враховуючи мою власну неприязнь до пітону, а точніше до його форматування коду, можна зробити висновок, що найкраще код електронної компоненти розробляти саме за допомогою Arduino IDE C/C++.

Щоб уникнути проблем пов’язаних з специфікою імплементації саме якоїсь платформи розробка буде проводитись за допомогою Arduino, а надалі код можна розширювати платформ-специфічними інструкціями.

Низька Вартість: найдешевшою опцією, яка підходить для проекту, тобто містить необхідний мінімум, є Arduino nano v3 що базується на мікроконтролері ATmega328p так коштує зазвичай до 2 доларів, потрібно бути уважним оскільки існують плати які маркуються так само але базуються на мікроконтролері 168P в якому дуже мало пам’яті, якої точно буде недостатньо.

Достатній розмір пам'яті: хоча пам’ять у 328p досить обмежена і має лише 32kB (2kB зайнято бутлоадером) та 2kB SRAM, цього повинно бути достатньо для того щоб розмістити код та зміні для управління LoRa, компас та GPS та додатковий код.

Закладені обмеження: недоліком є наявність лише одного серійного порту, а оскільки він зайнятий інтерфейсом комунікації з комп’ютером, то для комунікації з додатковими пристроями доведеться використати програмний серійний порт.

Необхідний мінімум: вісім аналогових портів вводу/виводу, що дозволяє підключення більш ніж одного логарифмічного перетворювача. Також наявний вихід для підключення опорної напруги і внутрішній 10 бітний АЦП, роздільної здатності якого достатньо для перетворення прийнятих значень з логарифмічного перетворювача.

Вищенаведені параметри задовольняють мінімальні вимоги для імплементації сенсорної частини системи. У випадку ж приймальної частини системи, частина для отримання даних LoRa, цей пристрій буде використовуватись лише як проксі для ретрансляції повідомлень отриманих по протоколу LoRa до пристрою, що буде містити програмне забезпечення візуалізації отриманих даних. Тому особливі вимоги до цього пристрою системи — відсутні.

Такий підхід дозволить пізніше змінювати платформу на більш потужнішу без особливих проблем, оскільки кожен розробник модулів для

Arduino IDE намагається досягти найбільшої сумісності саме з платами Arduino, і зміна платформи буде лише полягати в додаванні кількох ifdef інструкцій в залежності від платформи для якої будуть компілюватись сирці. Прикладом такої зміни може бути наприклад більша розрядність АЦП в деяких мікроконтролерах, які зазвичай за замовчуванням сконфігуровані на сумісний 10 бітний режим, а вже спеціалізованими інструкціями можна обрати іншу розрядність. Також вибір платформи з обмеженою пам'яттю, буде спонукати до розробки оптимізованого коду, а витік пам'яті буде легше зафіксувати, ідентифікувати і проаналізувати, що дуже актуально для розробки IoT пристроїв.

## 1.2 Аналіз технологій передачі даних на велику відстань та аналіз радіочастотних діапазонів

Для того щоб забезпечити високу портативність пристрою, що розробляється, він повинен бути обладнаний бездротовим зв'язком, щоб його можна було встановити на відстані від оператора, звісно, що чим більша відстань тим краще, а для цього потрібно використати одну з технологій передачі даних на велику відстань. Визначитись з радіочастотними діапазонами та перевірити наскільки таке використання відповідає чинному законодавству, що й буде зроблено у цьому розділі.

### 1.2.1 Технології зв'язку середньої дальності

Розглянемо для прикладу дві найрозповсюдженіших технології, що призначені для передачі даних на відносно велику дальність це опенсорсний протокол OpenLRS та пропрієтарний протокол LoRa:

- OpenLRS використовує частоти в діапазонах 433MHz та 915MHz і забезпечує радіус дії в кілька кілометрів, зазвичай використовується в пультах керування [30]. За допомогою використання більш потужних радіо-модулів чи підсилювачів можна досягнути відстані до 10км. Недоліком використання більш потужних передавачів, є те що такий пристрій легко виявити.

- LoRa використовує інтерфейс з меншою швидкістю прийому-передачі, і використовує значно більший діапазон частот, в межах 433MHz, 868MHz і 915MHz. LoRa за рахунок технології широкосмугової модуляції забезпечує зв’язок до 10км при цьому зберігаючи малу потужність споживання, за рахунок малопотужного випромінювання [8]. Потужність випромінювання на пряму залежить від комбінації швидкості передачі даних та від розміру пакету [31].

Саме LoRa є найбільш розповсюдженою технологію, що використовується в IoT пристроях, вона набула такого розповсюдження за рахунок малого споживання струму та великої дальності передачі даних і дозволяє розмістити датчики на великій відстані від ретранслятора чи пристрою керування.

### 1.2.2 Частотні діапазони БПЛА

Розглянемо частотні діапазони які нам необхідно виявляти. Тобто це найбільш розповсюдженіші БПЛА, а засоби протидії їм мають відповідне покриття.

Більшість комерційних квадрокоптерів працюють в частотах 2,4GHz та 5,8GHz [32], такі чіткі значення використовуються лиш для короткого позначення діапазонів. І на прикладі DJI 3 це 2.400-2.4835GHz, 5.725-5.850GHz [33]. Autel використовують ті самі частоти але додатково в їх новій версії з’явились 5,2GHz та 900MHz, збільшення діапазону частот значно підвищило надійність зв’язку але наявність програмних багів пов’язаних з реакцією на

GPS сигнал призводить до їх втрати і відповідно вони широкого розповсюдження не набули [34].

У випадку ж FPV різноманітність пристроїв управління, та передачі відео призводить до ще більшого різноманіття частот, що використовуються. Тут і присутні притаманні пропрієтарним квадрокоптерам 2,4GHz, 5,8GHz, до них додаються системи контролю так як опенсорсна ELRS, чи пропрієтарна Crossfire, технології передачі відео, що використовують 900MHz, 1200-1300MHz, 3300MHz [35]. Але останнім часом придбати 3,3GHz майже неможливо, а 900MHz важко і вони дорогі, виробники просто схоже адаптуються під запити ринку [36]. Приклад покриття в діапазоні більше 5GHz різними передавачами відео зображено на рисунку 1.1 [37].

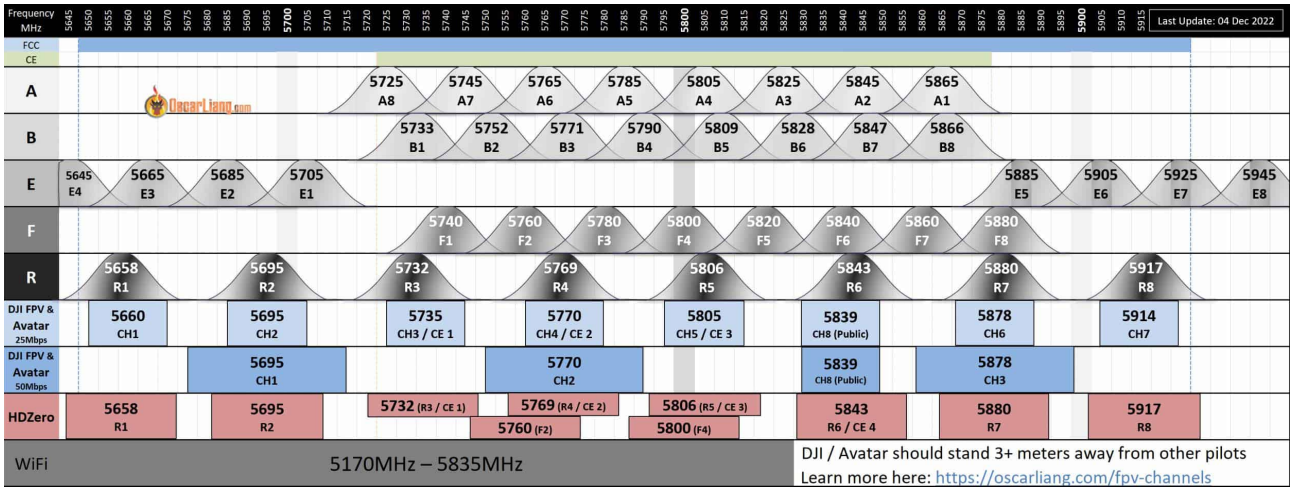



Рисунок 1.1 — Частотні діапазони відео передавачів FPV дронів

Також необхідно згадати і військові ворожі БПЛА такі як Орлан, Ланцет та Зала. У випадку Орланів і Ланцетів які використовують 800-900MHz ці частоти відомі [38], мало того вони використовують LoRa, саме так їх наявність і визначає пристрій «Цукорок» [39]. Зала ж додатково використовує 2,4GHz, яка спостерігається також і в ланцетах новіших версій. Як видно нічого нового вони

не використовують і там всі ті ж самі частоти та технології, які в основному розроблені і виробляються компаніями країн демократичного світу.

### 1.2.3 Законодавство розподілення частот

Оскільки наш пристрій передбачає передачу інформації в радіочастотному діапазоні. Потрібно перевірити чи дозволене використання частот заданої потужності згідно Національної Таблиці Розподілу Смуг Радіочастот України, що затверджено кабінетом міністрів України [40], а також відповідає регламенту аматорського радіозв'язку України [41, 42]. Перевіримо частоти найнижчого діапазону LoRa та знайдемо «Характеристики аматорських систем для амплітудної маніпуляції Морзе, PSK31, NBDP та режимів зі слабким сигналом, які працюють на частотах нижче 900 МГц», що на частотах 420 — 450MHz, обмеження потужності передавача 3-31,7dBm, та коефіцієнт підсилення антени в межах -3 — 23 dBi.

Отже, у випадку коли потрібне невелике споживання струму, велика відстань передачі даних і немає необхідності передавати великі об'єми даних, та мінімальний радіочастотний слід то найкращою технологію, на даний момент є LoRa. Оскільки LoRa має кілька імплементацій чіпів, які відрізніться частотою комунікації, та потужністю передавача, ми можемо обрати найкращий варіант для кожного випадку.

Так як основне призначення пристрою знаходження напрямок до місцеперебування джерела випромінювання таких як БПЛА або пристрою, що намагається заглушити комунікацію того самого БПЛА, то цей пристрій звісно буде використовувати той самий діапазон частот. Більшість БПЛА використовують вищезгадані частоти в вищому діапазоні радіочастотного спектру. Тому ми можемо обрати з лінійки LoRa модулів, той що найбільше

відповідає нашим вимогам, а саме якомога нижча частота передачі даних, щоб сам передача даних пристрою не впливала на свою ж роботу детекції радіочастотного випромінювання, це є SX1278, постійна потужність випромінювання, згідно даташиту не більше +20dBm для 3,3V, що цілком відповідає таблиці частот і значно нижче вказаного рівня в +31dBm [43, 44].

## 1.3 Аналіз фреймворків для розробки програмного забезпечення

В сьогоденні коли мобільні платформи набули широкого розповсюдження є просто неможливо ігнорувати їх наявність під час розробки нового проекту, а це як мінімум дві найпоширеніших платформи iOS та Android. При цьому необхідно не забувати про існування десктопних операційних систем як Windows від Microsoft, macOS від Apple та просто неймовірну кількість Linux дистрибутивів. Майже кожна операційна система має своє рідне середовище розробки. Тому необхідно розглянути доступні фреймворки для розробки програмного забезпечення.

Розглянемо їх відповідно до розповсюджених операційних систем:

1. macOS та iOS мають рідне середовище розробки це Xcode та SwiftUI. Мови розробки Swift та Objective-C. Xcode – це офіційне середовище розробки для macOS та iOS. Він підтримує Swift, а також Objective-C для забезпечення сумісності зі старішими версіями macOS. SwiftUI — підтримує лише Swift для побудови інтерфейсів [45, 46]. Обидві мови є розробками Apple. У випадку розробки Objective-C програми можуть бути лише зкомпільовані на продуктах Apple. У випадку Swift хоч Apple і зробило його опенсорсним, широкого розповсюдження він не набув, і максимум де ці програми можна якось зкомпілювати це буде Linux або Windows але Android хоч і має компілятор але такий додаток просто не зможе працювати. Тому вибір таких мов та середовищ розробки значно обмежує платформи на яких можна запускати клієнт.

2. Android має своє середовище розробки Android Studio – це офіційне середовище розробки. Воно підтримує як Java, так і Kotlin. Kotlin тепер є рекомендованою мовою розробки для Android, хоча Java все ще широко використовується [47]. Тре зауважити, що хоч мову називають Java вона є насправді Java-like мовою, ця мова містить виклики Android API, і такий код не може бути запущений напряму в Java інтерпретаторі, приблизно те саме справедливе і для Kotlin. Тобто вибір такого середовища розробки і мови обмежує платформу і це є лише Android.

3. Windows має дві основних технології для розробки додатків це рідний Visual Studio який підтримує C++ та C# (за допомогою якого можна створювати UWP додатки). Програми написані на C++ компілюються лише для Windows і їх запуск лише можливий через емулятори API windows на Linux та macOS, але такий спосіб запуску зазвичай має проблеми з програмами що працюють з пристроями, і абсолютно відсікає всі мобільні платформи. У випадку C# (та інших .NET мов) умовою виконання є наявність .Net фреймворку, який останнім часом розробляється для Linux та macOS, але насправді є важким і досить недоробленим продуктом, і знову є таки відсутність підтримки на Android та iOS. Також за допомогою цих мов додатки можуть бути розроблені для UWP платформи і хоча інтерпретатором таких програм працює .NET, UWP додатки не можуть виконуватись на інших платформах і обмежені лише запуском на Windows пристроях [48].

Як видно жоден з рідних способів розробки від виробників операційних систем не забезпечує необхідної портативності. Тому розглянемо кілька альтернатив платформонезалежної розробки. Ми не будемо розглядати інтерпретовані мови типу React чи Javascript оскільки навіть коли вони зібрані для мобільних платформ вони використовують інтерпретатор, який поставляється менеджерами пакетів такими як npm і де постійно знаходять пакети з вбудованими троянами і тому їх не можна вважати безпечними [49].

Найбільш популярна на даний момент Java, яка є платформонезалежною, але має дуже застарілий модуль для побудови UI інтерфейсу, в якому більшість методів керування елементами вікна необхідно реалізовувати вручну. І вона потребує інтерпретатора передкомпільованого байтерейкоду, що знову ж таки унеможливлює запуск на iOS, Android без значної модифікації коду [50, 51].

Qt фреймворк з відкритими сирцями який початково розроблявся компанією Trolltech, був куплений Nokia яка гальмувала його розвиток і на даний момент підтримується компанією The Qt Company. Підтримує дуже великий спектр як операційних систем так і процесорів на яких вони виконуються, саме тому цей фреймворк використовується багатьма автомобільними виробниками для специфічних бортових комп’ютерів. Написані програми компілюються в рідний код для кожної платформи. Мови програмування C++ та декларативний QML. Даний фреймворк, часто забезпечує навіть роботу з апаратними пристроями на рівні свого API, що дозволяє розробнику абстрагуватись на специфічності роботи з певною платформою. Компіляція для iOS відбувається на MacOS з встановленим XCode в результаті чого, на виході отримується рідний ipa додаток. Так само компіляція для андроїд потребує встановлених Android SDK/NDK, в результаті чого буде зкомпільовний рідний apk додаток [52].

Як видно з вище викладеного Qt у нашому випадку, коли основна вимога не бути обмеженим платформою, є найкращим вибором, що дозволить створити додатки як для настільних, так і для мобільних платформ, а також дозволить компілювати додаток навіть для вбудованих пристроїв за умови підтримки їх процесора компілятором.

## 1.4 Аналіз параметричних 3d редакторів

Оскільки для даного пристрою доведеться розробляти деталі не найпростішої складності необхідно розглянути наявні параметричні 3D редактори, що і буде зроблено в цьому розділі.

Найбільш популярним параметричним 3D дизайнером серед професіоналів є SolidWorks, дійсно з великими можливостями розробки, але ціна його біля 360$ на рік [53, 54]. Тому для звичайної людини яка не заробляє цим на життя не має сенсу в купівлі такого продукту.

OpenSCAD є параметричним 3D редактором з відкритими сирцями, але для створення об’єктів в ньому використовується своя мова, так вона дозволяє робити досить цікаві речі, шляхом математичних розрахунків, але створення таких простих речей як різьби займає більше часу ніж те необхідно. Та і вивчення мови потребує додаткового часу [55].

FreeCAD теж є опенсорсним 3D редактором, але на відміну від OpenSCAD він за своєю ідеологією більше схожий на SolidWorks. В багатьох випадках має не гірший функціонал. Його основна проблема це баги в ядрі які навряд чи хтось колись виправить тому він може інколи падати, а також він стає дуже повільним, якщо у вас складна деталь з великою кількістю залежностей. Незважаючи на такі недоліки, у ньому доступно багато панелей які полегшують життя розробника і наближають його до професійного рівня. Те що він з відкритими сирцями дозволяє все таки виправляти баги за допомогою широкої спільноти користувачів. Параметричний дизайн дозволяє ітеративну розробку та тестування різних концепцій розробки пристроїв. В ньому присутні базові можливості роботи з сітками 3D моделей, хоча коли потрібна більш глибока не параметрична модернізація краще застосовувати редактори типу Blender, в свою чергу модель після такого редагування можна імпортувати FreeCAD і далі використовувати її як базу для подальшого параметричного моделювання.

Широка спільнота забезпечує доступність та оновлення модулів та скриптів, наприклад такий модуль як створення потрібної різьби, що відповідає необхідним стандартам [56, 57]. Основна панель розробника зображена на рисунку 1.2.

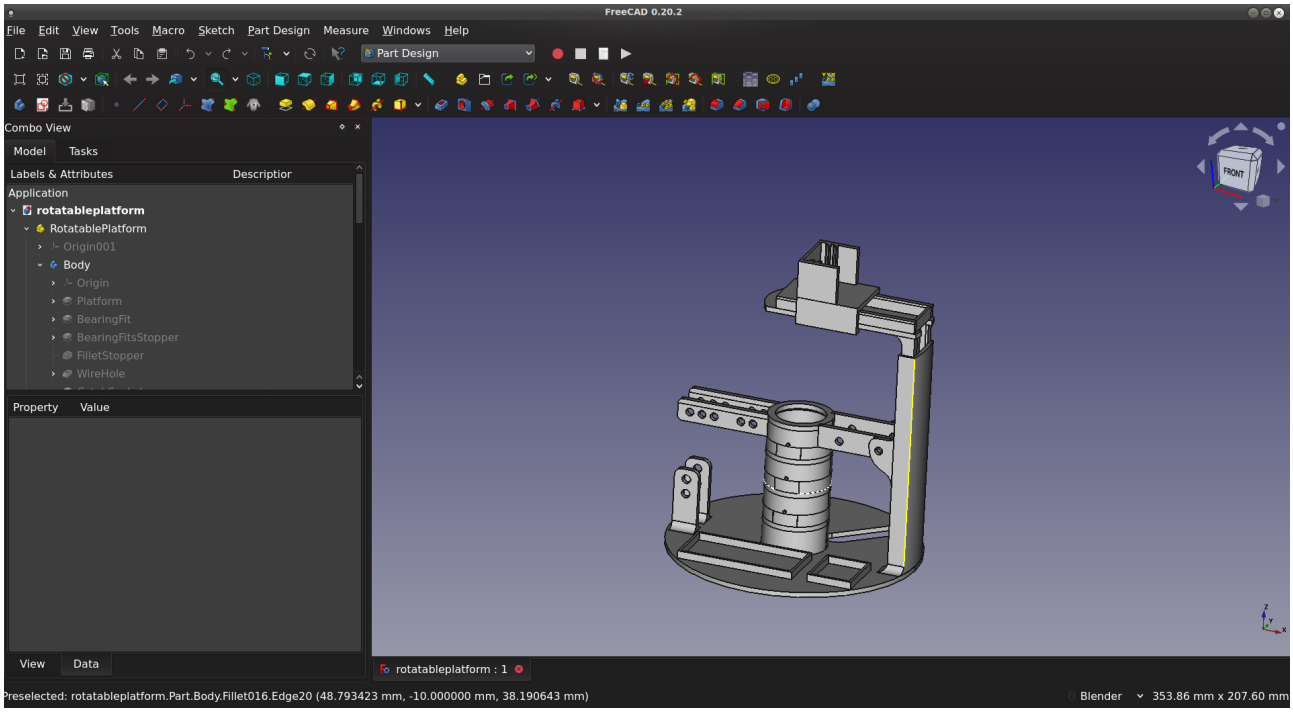



Рисунок 1.2 — Основна панель розробника у FreeCAD

Як видно з вищевикладеного для розробки доцільно використовувати саме FreeCAD який є безкоштовний і надає всі основні інструменти для створення досить складних пристроїв.

# 2 РОЗРОБКА СТРУКТУРИ СИСТЕМИ ВИЗНАЧЕННЯ ДЖЕРЕЛА РАДІОВИПРОМІНЮВАННЯ

На початкових стадіях розробки необхідно розробити структурну схему системи. Визначити основні функціональні частини системи, їх призначення та взаємозв'язки, що і зроблено в цьому розділі.

Дана система складається з таких основних компонентів:

1. Пристрій-турель для реєстрації радіочастотного випромінювання, збору цих даних та відсилання їх за допомогою LoRa до приймача;
2. Пристрій-приймач виконує роль проксі приймача даних з інтерфейсом LoRa та передачу його на станцію візуалізації даних;
3. Програма для візуалізації даних, персональний комп’ютер або мобільний пристрій з програмою яка візуалізує дані з приймача LoRa.

Спрощена структурна схема взаємодії компонентів системи показана на рисунку 2.1.


Рисунок 2.1 — Структурна схема взаємодії компонентів системи

## 2.1 Структурна схема пристрою-турелі

Основною частиною даної системи є пристрій-турель що буде використовуватись для вимірювання потужності радіовипромінювання та його напрямку.

Для вирішення ряду задач поставлених перед цим пристроєм, а саме збір показників потужностей, напрямок джерела радіовипромінювання, місцеперебування пристрою. Розглянемо структурну схему такого пристрою рисунок 2.2.

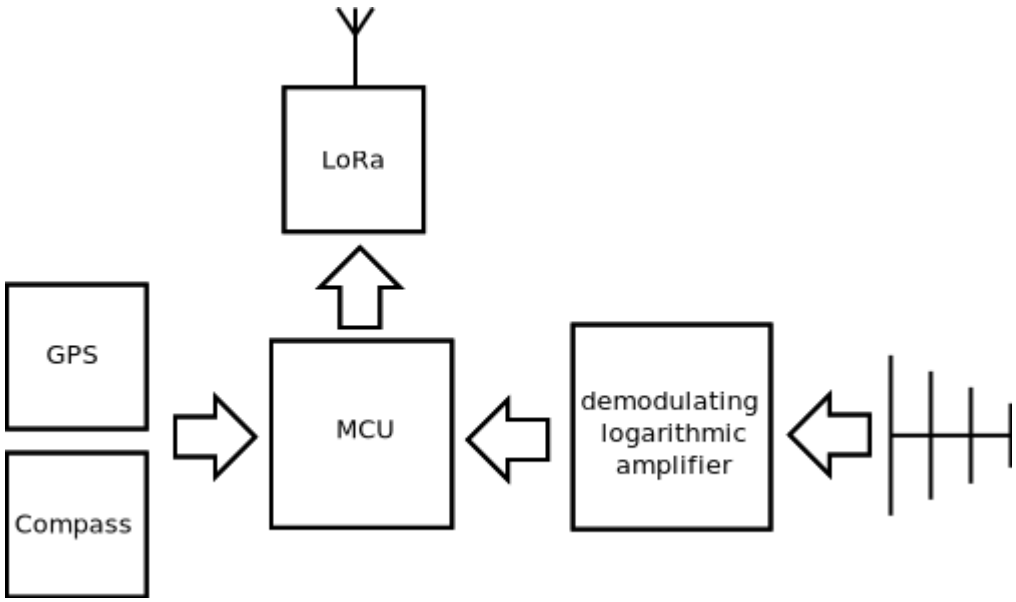



Рисунок 2.2 — Структурна схема пристрою-турелі

Обов’язковими компонентами такої системи є напрямна антена, що приймає сигнал, сигнал подається на демодулюючий логарифмічний підсилювач, що перетворює рівень сигналу у відповідне значення напруги, компас для визначення напрямку, та GPS  модуль для реєстрації координат

положення самого пристрою. Після відповідної попередньої обробки, формування пакетів, дані транслюються по протоколу LoRa.

## 2.2 Схема електрична принципова пристрою-турелі

Пристрій-турель складається з двох частин: статична основа та обертальна, оскільки якість роботи багатьох компонентів залежить від їх розташування на корпусі, система повинна в більшості своїй бути модульною, з можливістю рознесення на корпусі і розподіленого монтажу. Загальний вигляд схеми електричної принципової наведено на рисунку 2.3.

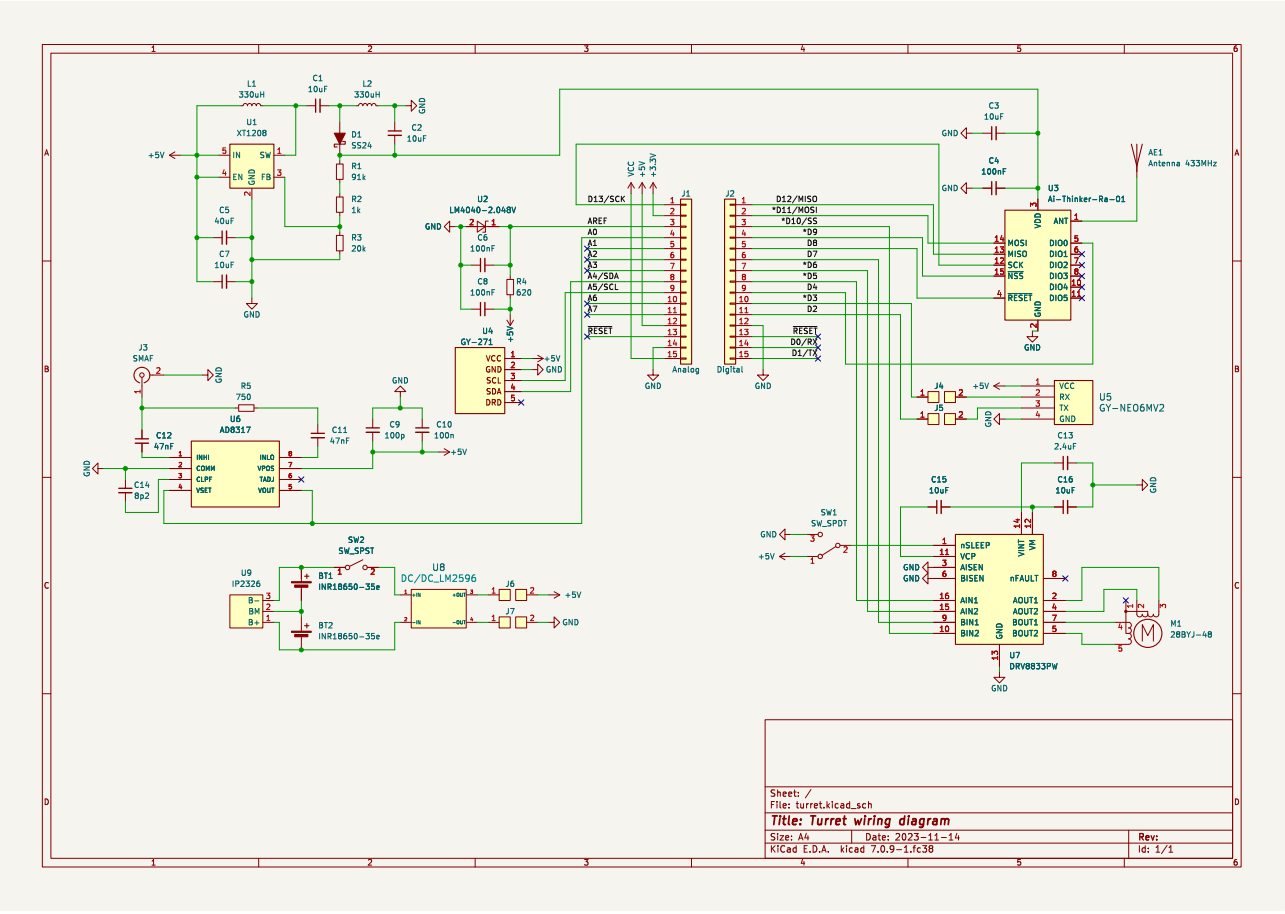



Рисунок 2.3 — Схема електрична принципова пристрою-турелі

Дві компоненти турелі, рухома та нерухома з’єднанні контактами-підшипниками J4, J5, J6, J7. Живлення системи здійснюється за допомогою двох 18650 li-ion елементів BT1 та BT2, сумарною напругою 8,4V. Заряд елементів живлення контролюється платою U9 контролю заряду IP2326, що дозволяє заряджати пристрій під’єднавши до порту USB. В свою чергу DC-DC перетворювач LM2596 виконує перетворення 8,4V до рівня 5V необхідного для живлення компонентів пристрою.

Для забезпечення надійної і швидкої фіксації GPS сигналу, та подальшого приймання стабільного сигналу, модуль GPS U5 GY-NEO6MV2 має знаходитись на нерухомій частині турелі, оскільки обертання негативно впливає на роботу модуля.

Спілкування мікроконтролера з модулем GPS відбувається по UART протоколу. GPS модуль приєднаний до серійного порту мікроконтролера за допомогою підшипників-контактів J4 та J5, дрижання контактів лінії нівелюється шляхом паузи в обертанні на час передачі даних. У нашому випадку буде використовуватись програмний серійний порт, незважаючи на недоліки такого підходу, у нашому випадку це не є критичним тобто GPS RX, TX заходять на входи мікроконтролера D3, D2 відповідно. Живлення як найважча по масі частина пристрою, теж розміщується на статичній частині, і шляхом вмикання SW2, через підшипники-контакти J6 та J7, потрапляє на рухому частину пристрою-турелі.

Обертання рухомої частини турелі здійснюється за допомогою крокового двигуна 28BYJ-48 M1 з напругою живлення 5V, контроль мотора виконаний на мікросхемі DRV8833 U7 [58], це подвійний H-bridge побудований на N канальних MOSFET транзисторах, що забезпечує гарну енергоефективність, а за необхідності дозволяє обмежити струм споживання мотором, у нашому така

необхідність відсутня тому xISEN контролера U7 під’єднані до GND. Керування драйвером здійснюється за допомогою цифрових ліній мікроконтролера D5, D6, D7, D10 відповідно приєднаних до входів драйвера AIN1, AIN2, BIN1, BIN2. Драйвер мотора підтримує режим сну, який використовується для вимкнення мотору і зупинки рухомої частини пристрою, для можливості зміни прошивки, оскільки контролер знаходиться на рухомій частині. Тобто шляхом перемикання SW1 можна увімкнути обертання  подавши +5V на вхід nSLEEP, чи призупинити обертання турелі перемкнувши на перемикач на землю.

Мікроконтролер зчитує дані про напрямок повороту турелі з акселерометра GY271 U4 з функцією магнітометра, тобто компаса. Який в свою чергу приєднаний до синхронної шини I2C, виходи SDA та SCL до виходів мікроконтролера A4,A5. Важливо щоб магнітометр знаходився якнайдалі від мотора, який може впливати на вимірювання показників напрямку.

Безпосереднього для вимірювання потужності радіовипромінювання використаємо AD8317 U6 це демодулюючий логарифмічний підсилювач, що здатен з високою точністю конвертувати вхідний сигнал у відповідне децибел-скальоване значення напруги на виході. AD8317 забезпечує акуратну логарифмічну відповідність для сигналів від 1MHz до 8GHz, і має прийнятну точність аж до 10GHz. Струм споживання зазвичай не більше 22mA. Вхідна потужність не повинна перевищувати 12dBm, оскільки ми використовуємо антену, то цей показник перевищити важко [59]. Оскільки у нашому випадку нам не потрібне дуже точне значення потужності ми можемо знехтувати $R_{tadj}$, для подальшого спрощення схеми, і тому вихід TADJ мікросхеми можна залишити не під’єднаним, тобто відкритим згідно документації. Саме з тієї причини ми не використовували в нашій схемі температурний датчик, незважаючи на те що можна було б коригувати програмно виміряні показники відносно температури. Використано стандартну конфігурацію підключення VSET та VOUT рисунок 2.4.

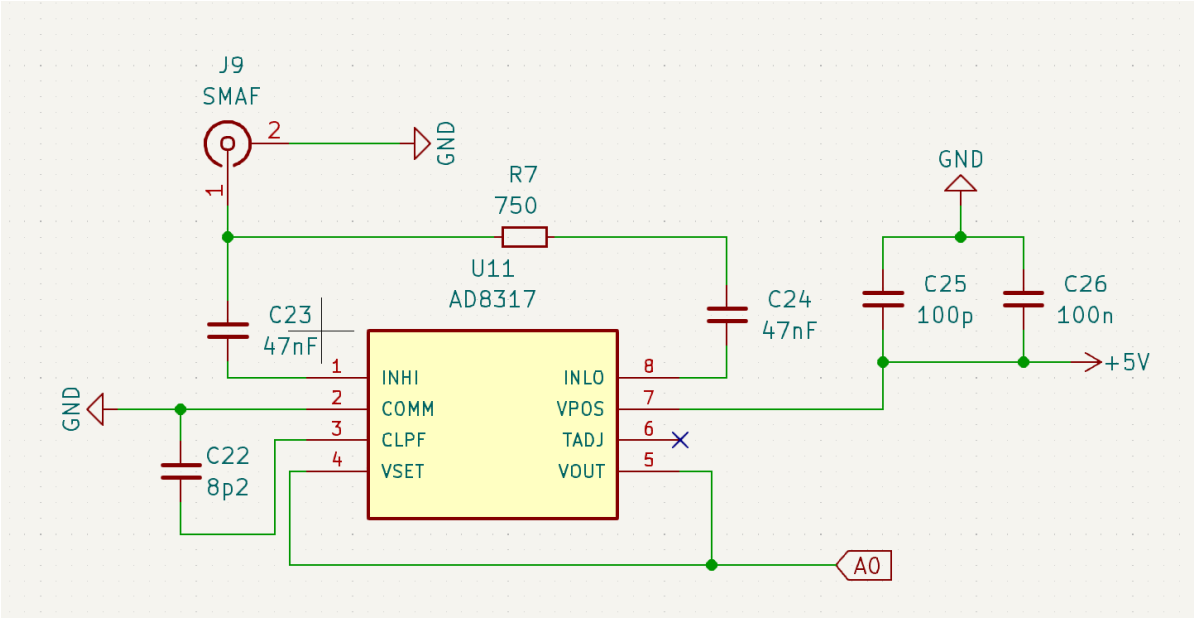



Рисунок 2.4 — Схема включення логарифмічного демодулюючого підсилювача AD8317

Тобто VSET напруга виникає лише на внутрішньому резисторі 1,5kOhm викликаючи зворотній зв’язок VSET інтерфейсу, і не викликаючи зміни номінальної логарифмічної похилої. Яка за даної конфігурації дорівнює приблизно -22 mV/dB з відповідно приблизним відхиленням в діапазоні від 0,35V до 1,7V. Саме цей сигнал подається на вхід мікроконтролера з АЦП A0 де буде проводитись  подальша обробка сигналу, що приймається. Залежність вихідної напруги до вхідного сигналу AD8317 є лінійною залежністю відскальованою до dBm і описується формулою 2.1.

$$V_{OUT} = X \times V_{SLOPE/dB} \times 20 \times \log_{10}(V_{IN}/V_{INTERCEPT}) \qquad (2.1)$$

де $X$ - це фідбек фактор $V_{SET}=V_{OUT}/X$,

$V_{SLOPE/dB}$ - у випадку нашого підключення -22mV/dB,

$V_{INTERCEPT}$ - виміряна частина напруги, представлена як лінійна децибел залежність від $V_{OUT}$.

Отже виходячи з вище викладеного на вихід мікроконтролера який контролює роботу пристрою буде надходити напруга в межах 0,35-1,7V. Необхідно обрати значення опорної напруги якомога ближче за значенням до найбільшого значення що буде вимірюватись, щоб отримати найбільшу градацію за у мови використання внутрішнього 10 бітного аналого-цифрового перетворювача. Зазвичай Arduino плати забезпечують лише 5Vабо 3,3V опорну напругу, що у нашому випадку забагато, а аналоги, що мають меншу внутрішню опорну напругу зазвичай не дуже надійні і стабільні, а оскільки ця напруга внутрішня то аналіз проблем значно ускладнюється. Щоб уникнути даних проблем доцільніше використовувати джерело зовнішньої опорної напруги. В схемі використано точний мікропотужний шунт опорної напруги LM4040. Найменша доступна напруга в даній лінійці мікросхем це 2,048V це значення цілком задовольняє наші потреби перекриває максимальне значення 1,7V з деяким запасом. Використаймо LM4040A20I U2, тобто це чіп грейду А з максимальним відхиленням опорної напруги 0,1%. Для того щоб використати стандартну схему підключення шунт-регулятора необхідно розрахувати опір R4, що включається між живленням та катодом LM4040 [60]. Максимальний струм що може забезпечувати регулятор не більше 15mA і розраховується за формулою 2.2

$$R_S=\frac{(V_S-V_Z)}{(I_L-I_Z)} \quad (2.2)$$

де $V_S$ - напруга живлення,

$V_Z$ - зворотня напруга пробою,

$I_L$ - струм навантаження,

$I_Z$- струм катоду.

Напруга живлення 5V, зворотня напруга пробою для LM4040A20I 2,048V, струм навантаження це необхідний струм для Iref Arduino біля 5mA, візьмемо мінімальний струм катоду за основу з документації LM4040A20I це 45uA. Для нашого випадку отримаємо

$$R_S=(5-2{,}048)/(0{,}0048+45\cdot10^{-6})=609 \text{ Ohm}.$$

Отже можна використати резистор 620 Ohm, що буде забезпечувати приблизно 4,8-4,9mA, а це і є необхідний струм для роботи.

Сформовані пакети транслюються за допомогою LoRa модуля Ra-01 U3. Комунікація мікроконтролеру та модуля відбувається за допомогою SPI інтерфейсу, тому виходи модуля MOSI, MISO, SCK, приєднанні до відповідних виходів мікроконтролера D11, D12, D13. NSS вихід модуля до D9 модуля мікроконтролера, Reset до D8, DIO0 до D4. Конденсатори C3 та C4 рекомендований спосіб стабілізувати живлення модуля від 3,3V. Оскільки модуль використовується для передачі даних то згідно документації типове використання потребує струм 140mA і рекомендовані вимоги до джерела живлення, це здатність забезпечувати більше 200mA, оскільки це значення, що

може сягати споживання LoRa модуля в пікових значеннях [61]. Також відомо що максимальний піковий струм, що може забезпечувати Arduino модуль зазвичай не перевищує 150mA. Тому для універсальності конструкції використано зовнішнє джерело напруги 3,3V, на мікросхемі XT1208 U1 [62], що забезпечує струм більше 600mA достатній для стабільної роботи LoRa модулю.

Повний лист компонентів використаних для пристрою наведений в таблиці 2.1.

Таблиця 2.1 — Перелік компонентів пристрою-турелі

| **Part** | **Pcs** |
|---|---|
| Mini / Type-C / Micro USB Nano 3.0 With the bootloader compatible Nano controller for arduino CH340 USB driver 16Mhz ATMEGA328P | 1 |
| 5V Stepper Motor 28BYJ-48 | 1 |
| DRV8833 2Channel DC Motor Driver Module Board 1.5A 3V-10V | 1 |
| DC-DC Auto Boost Buck Converter Module | 1 |
| Ra-01 LoRa SX1278 433M | 1 |
| PCB 12dBi 2.4GHz WIFI Yagi Directional Antenna | 1 |
| AD8317 High Speed Logarithmic Detector | 1 |
| RG316 RG316 RPSMA M SMA M, 10CM | 1 |
| INR18650-35E batteries | 2 |
| Type-C USB 2S BMS 15W 8.4V Lithium Battery Charging Boost Module With Balanced Support Fast Charge With Indicator | 1 |
| LM2596HVS DC-DC Adjustable Step Down Buck Converter Power Module | 1 |
| GY-NEO6MV2 NEO-6M GPS | 1 |
| HMC5883 GY-271 3V-5V Triple Axis Tri-axis 3 Axis Compass Magnetometer Sensor | 1 |

В додатку Б наведено схему електричну принципову пристрою турелі.

## 2.3 Вибір вимірювальної антени та розробка відповідного методу визначення джерела радіочастотного випромінювання

В усій системі чи не найважливішу роль відіграє антена, що використовується для вимірювання. Останнім часом все збільшується кількість FPV, та частот, що ними використовуються, але все ж найбільш популярними залишаються 2,4GHz, 5,8GHz [63]. Більшість комерційних дронів мають вертикальну поляризацію антени. А оскільки до антени висувається вимога, щоб нею можна було визначити напрямок вона має бути напрямна. Також завдання вибору антени ускладнюється ще і розміром пристрою, щоб зберігати його портативність.

В даній розробці застосовано напрямну антену типу Ягі, поляризацію антени визначає її розташування в корпусі, також плюсом такої антени є те, що вона, хоч і з значними втратами в силі сигналу, буде приймати сигнал від джерел що використовують антени з циркулярною поляризацією, а це в основному передача відео від саморобних FPV, незалежно від їх напрямку поляризації чи то права чи ліва.

Обрана антена усього 130 cm в довжину та згідно заявлених характеристик має значний коефіцієнт підсилення в діапазоні 2.4G 12dBi та 5.8G 3dBi, та подвійний діапазон, з V.S.W.R <2 рисунок 2.5 та таблиця 2.2.

Таблиця 2.2 — Заявлені характеристики антени

| | |
|---|---|
| Model | A-001-2.4G |
| Frequency range | 2400~2500MHz / 5600~5900MHz |
| Gain | 2.4G 12dBi / 5.8G 3dBi |
| SWR | <2.0 |

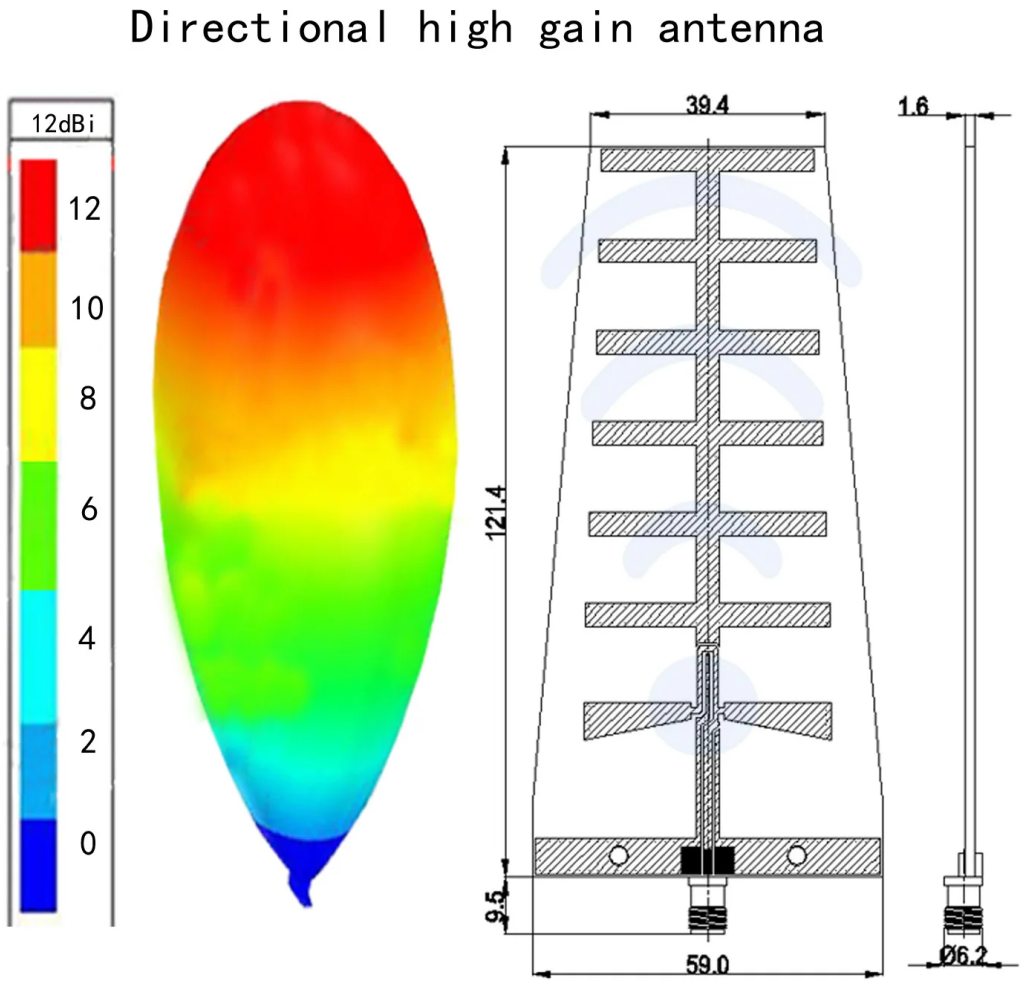



Рисунок 2.5 — Фізичні характеристики антени для вимірювання

Для того щоб переконатись, що антена відповідає нашим вимогам, проведемо її вимірювання, вимірювання, проводились з антеною не змонтованою та змонтованою в корпусі прототипі, показали що корпус не впливає на показники антени. Вимірювання проводились за допомогою Nano VNA та програми візуалізації NanoVNA saver.

Спочатку проведемо попереднє вимірювання у всьому спектрі діапазону частот, що нас цікавить, це 800MHz-5900MHz. Вимірювання показало, що найменша частота на який виникає резонанс знаходиться вище 1600MHz, а також резонанс виникає додатково ще на кількох частотах, тому проведемо додаткове попереднє вимірювання, але вже з початковою частотою трохи менше мінімальної частоти резонансу, вимірювання  наведене на рисунку 2.6.

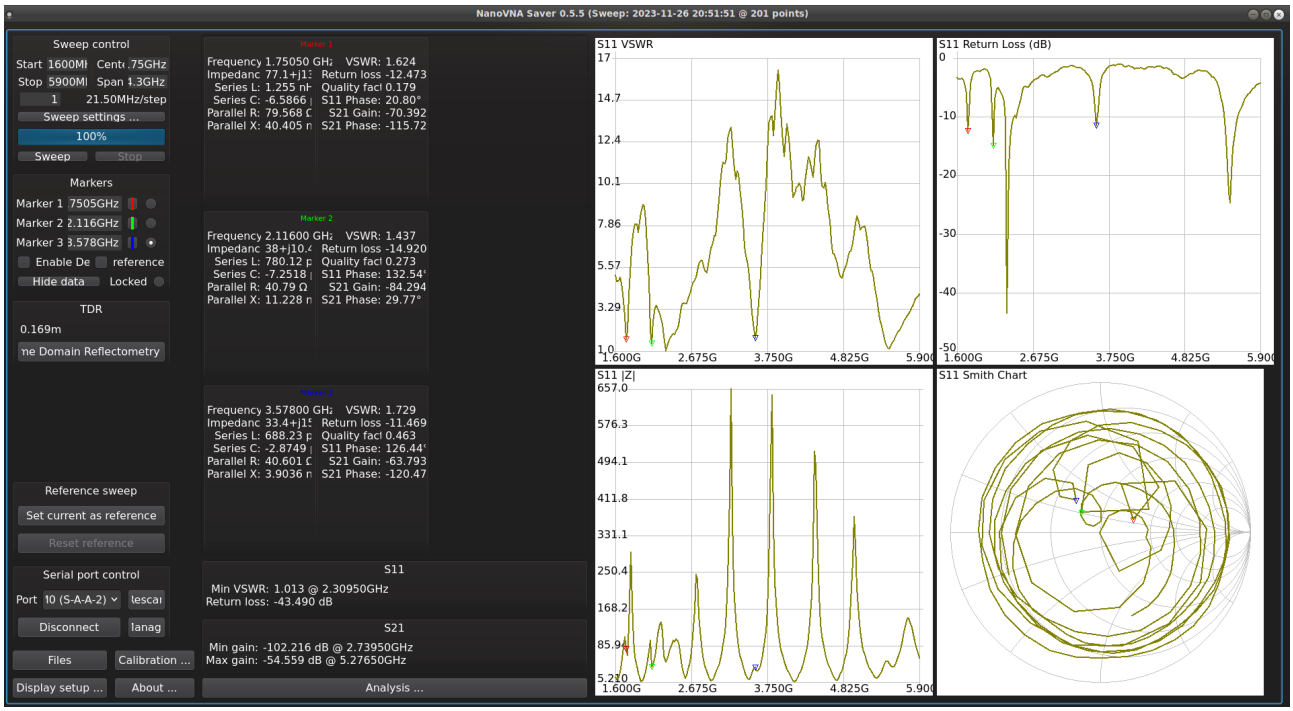



Рисунок 2.6 — Попереднє вимірювання антени у діапазоні частот 1600MHz-5900MHz

З рисунку 2.6 видно, що додатково резонанс виникає на частотах ~1,7GHz, ~2,1GHz, ~3,5GHz. Характеристики антени на цих частотах потрібно перевірити додатково до тих що заявлені в специфікації.

Вимірювання на частотах 1700MHz-1800MHz показало посереднє значення VSWR 1,6 з частотою резонансу 1758MHz, приблизним діапазоном частот 50-80MHz і представлене на рисунку 2.7.

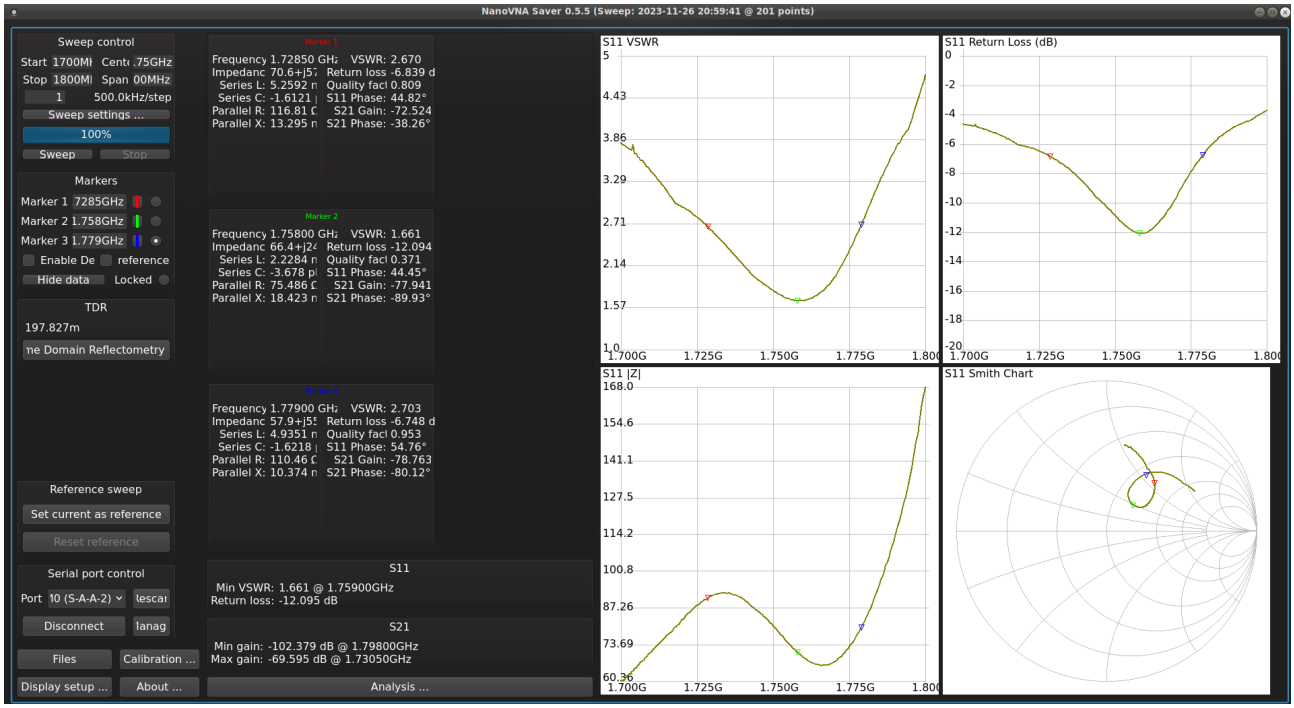



Рисунок 2.7 — Вимірювання антени на частотах 1700MHz-1800MHz

Наступний діапазон вимірювання 2050MHz-2150MHz з частотою резонансу 2110MHz хороший показник VSWR але досить невеликий діапазон приблизно в 50-60MHz. Вимірювання зображене на рисунку 2.8.

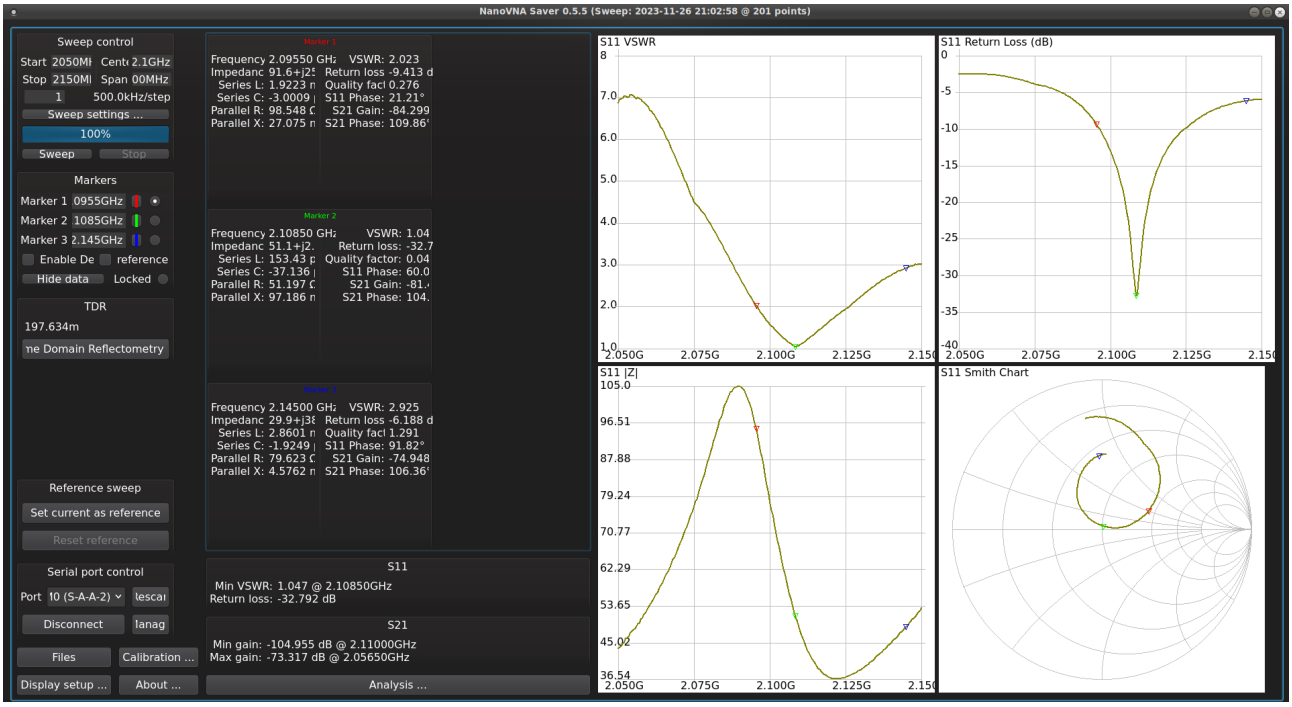



Рисунок 2.8 — Вимірювання антени на частотах 2050MHz-2150MHz

Наступне вимірювання проведемо для діапазону частот, що заявлено в специфікації 2250MHz-2500MHz, і це помітно. Досить широкий частотний діапазон навіть значно більший ніж заявлений виробником, а саме частота резонансу 2310MHz, діапазон майже 200MHz з VSWR <2, результати вимірювання показані на рисунку 2.9.

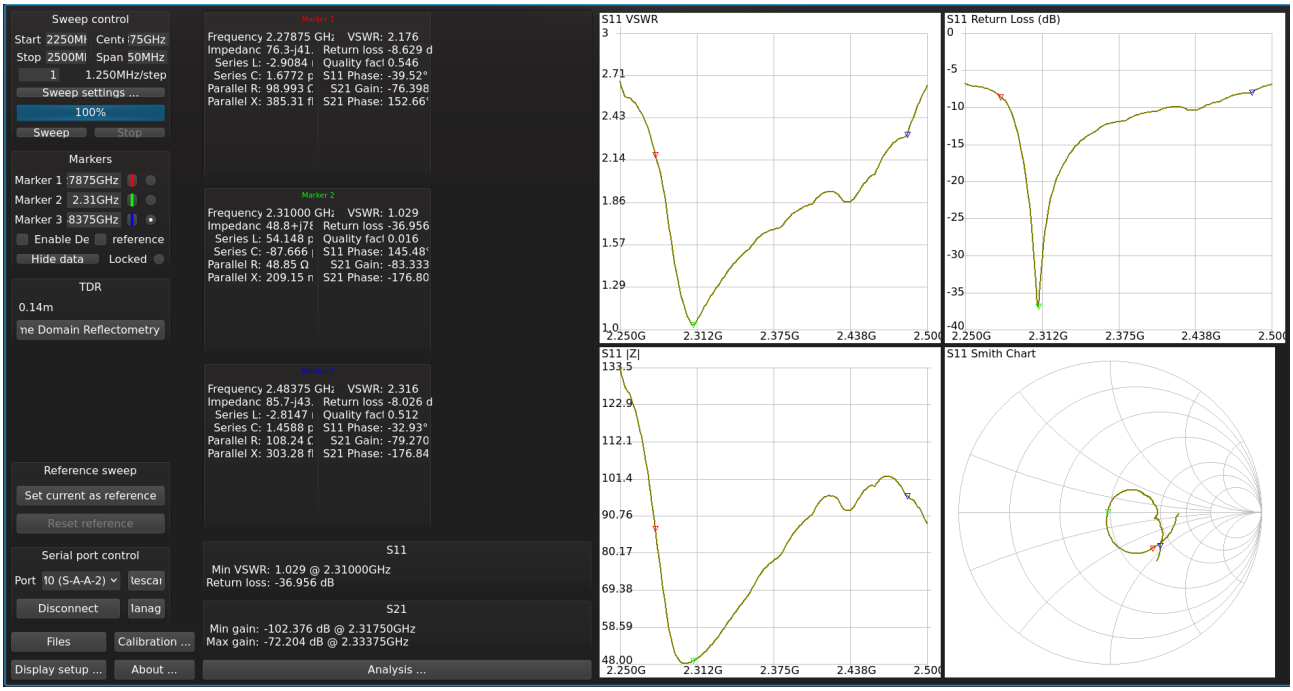



Рисунок 2.9 — Вимірювання антени на частотах 2250MHz-2500MHz

Останнє вимірювання на частотах, що не були заявлені виробником проведемо в діапазоні 3500MHz-3650MHz, діапазон досить широкий приблизно 100MHz з VSWR менше 3, вимірювання на рисунку 2.10 можливо виробник не включив всі недокументовані частоти в перелік із-за досить вузьких діапазонів з невеликим VSWR, а можливо просто тому, що вони рідко використовуються.

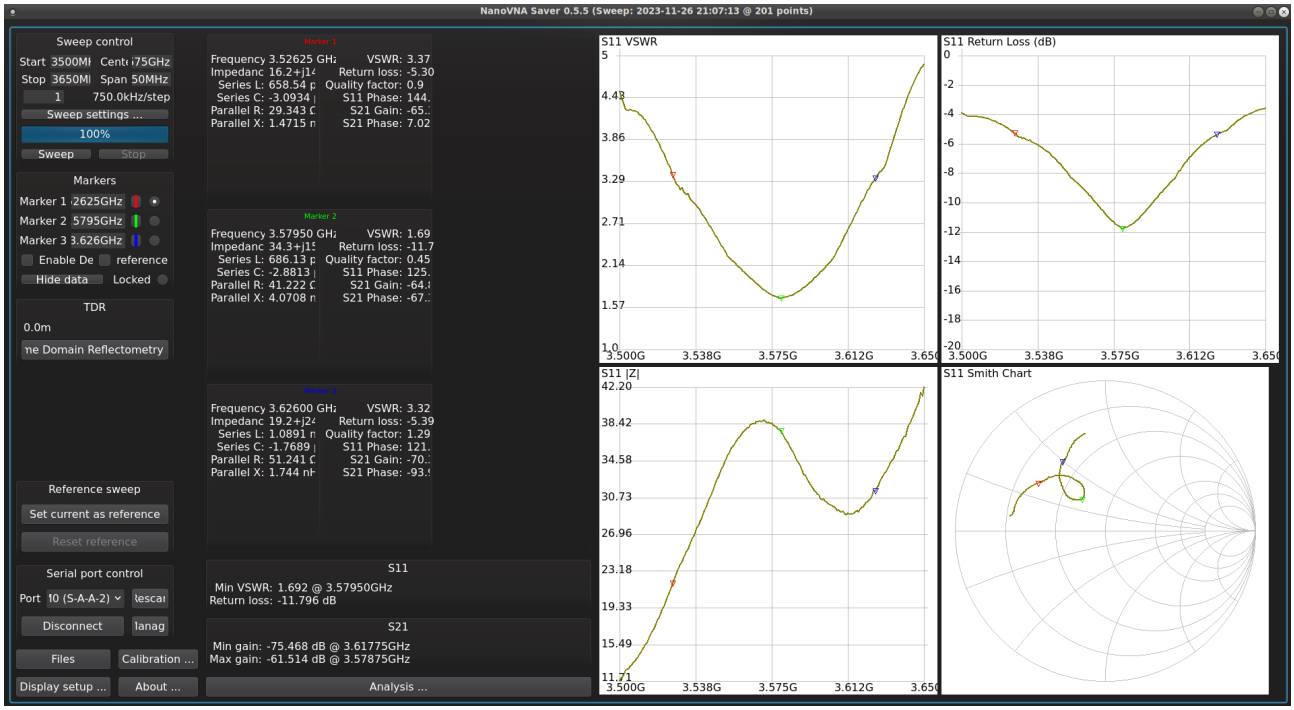



Рисунок 2.10 — Вимірювання антени на частотах 3500MHz-3650MHz

Проведемо вимірювання останнього діапазону частот заявленого виробником 5300MHz-5900MHz, як бачимо виробник трохи викривив показники антени і насправді, якщо зберігати його критерій VSWR менше двох, то ми отримуємо частотний діапазон 5400-5600MHz, звісно що звичайний споживач, зазвичай не проводить замірів, а навіть при VSWR 3+ антена якось буде працювати. Вимірювання на цьому діапазоні показано на рисунку 2.11.

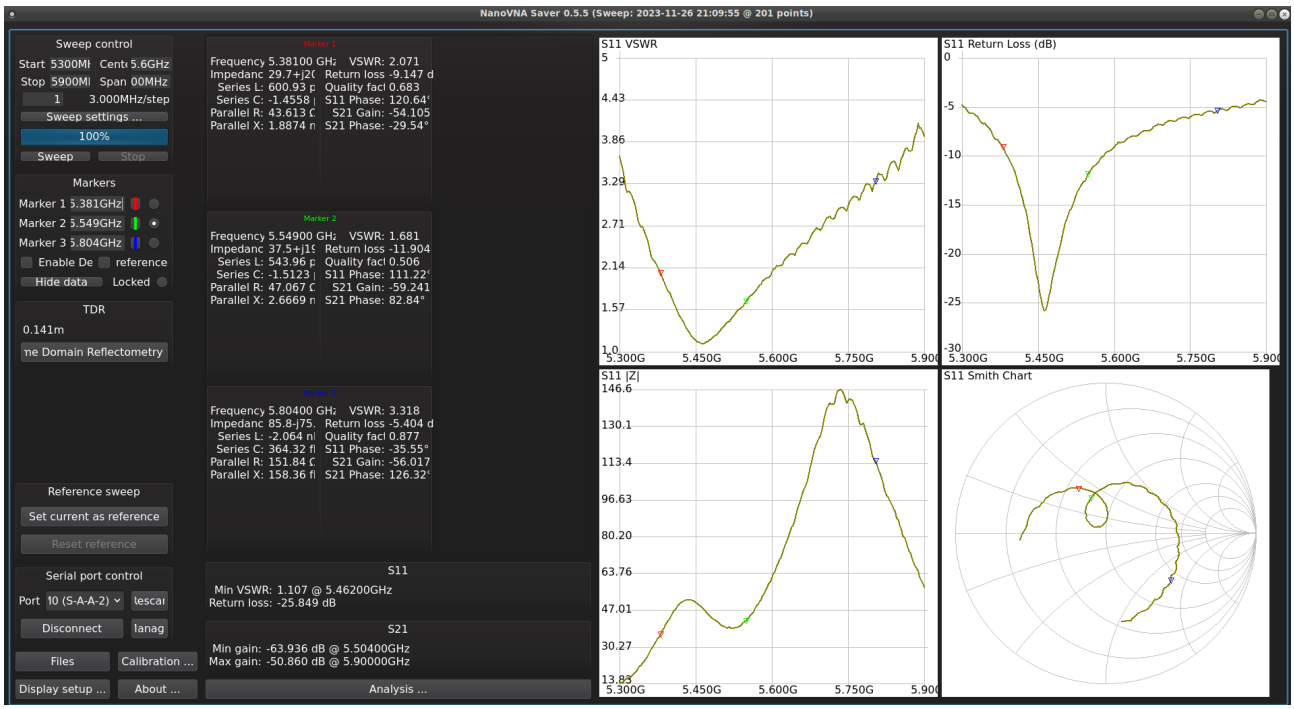



Рисунок 2.11 — Вимірювання антени на частотах 3500MHz-3650MHz

Оскільки вимірювальна антена і є визначальним елементом в якому діапазоні частот працює пристрій, то ці вимірювання і є показниками нашого пристрою. Отже робочий діапазон частот пристрою наведено в таблиці 2.3.

Таблиця 2.3 — Робочі частоти пристрою вимірювання частот

| Frequency Range (MHz) | VSWR | Min VSWR | Min VSWR Frequency(MHz) |
|---|---|---|---|
| 1720-1780 | <2,7 | 1,7 | 1760 |
| 2080-2145 | <2,9 | 1,05 | 2110 |
| 2250-2500 | <2,6 | 1,03 | 2310 |
| 3525-3625 | <3,3 | 1,7 | 3580 |
| 5310-5800 | <3,29 | 1,11 | 5460 |

Типовий патерн спрямованої вузьконаправленої антени типу Ягі, має краплеподібну форму і зображений на рисунку 2.12.

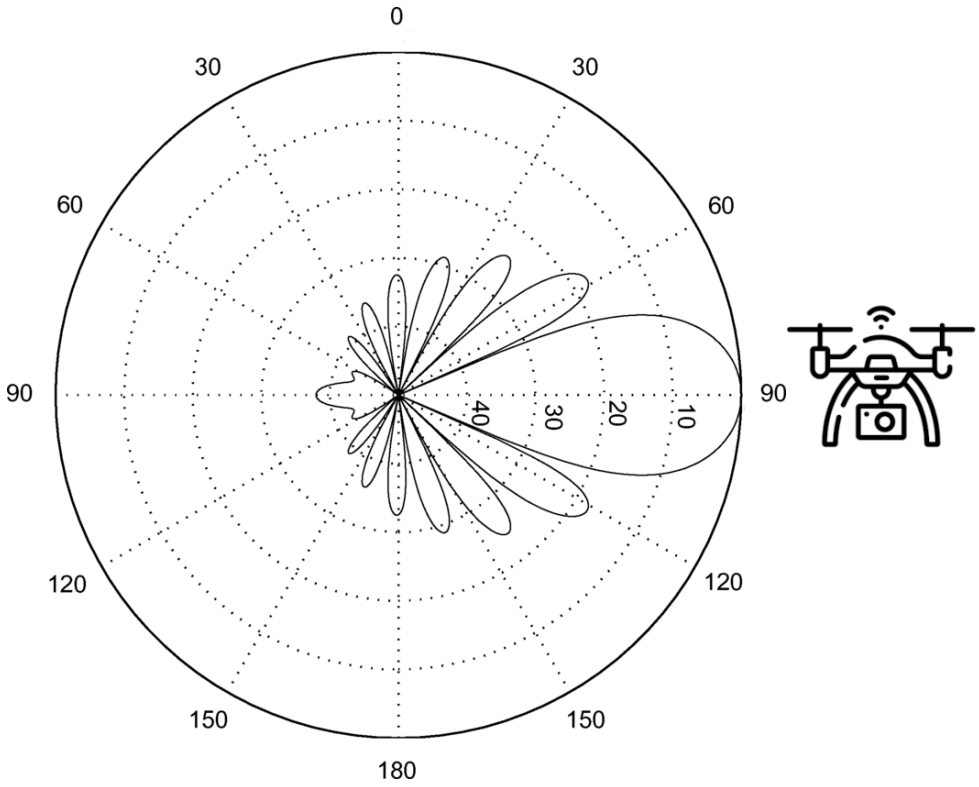



Рисунок 2.12 — Типовий патерн спрямованої антени

Отже як видно з рисунку для визначення джерела випромінювання можна використовувати метод пошуку максимуму. Метод полягає в безперервному обертанні антени допоки на антені не буде зареєстрований максимальна вихідна напруга, а отже напрямок антени збігається з напрямком реєстрації максимального значення напруги. Також потрібно врахувати, що UAV та інші цифрові джерела радіочастотного випромінювання не випромінюють постійно і на тій самій частоті, з однаковою потужністю, це можна побачити за допомогою

програмного радіо-аналізатора, на водоспадному графіку добре видно короткі сигнали, що продемонстровано на рисунку 2.13.

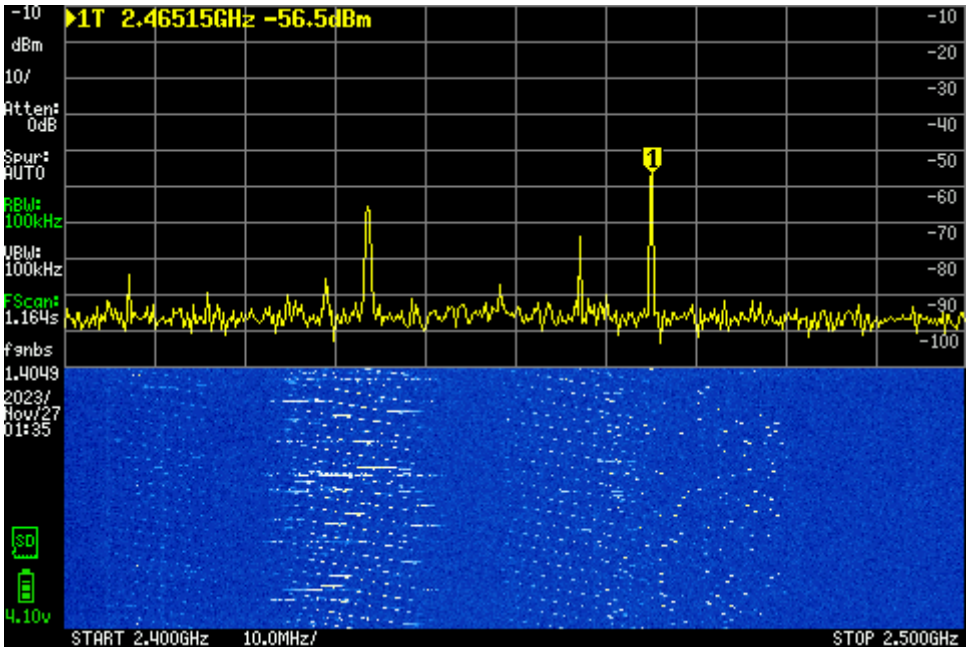



Рисунок 2.13 — TinySA сигнали Wi-Fi

Тому вимірювання потрібно проводити впродовж певного часу, зібрати кількість необхідних семплів і виділити серед них семпли з найбільшою напругою, тобто це можна представити формулою 2.3.

$$U_{max} = max(\sum_{i=1}^{n} U_i) \tag{2.3}$$

де $U_i$ - значення напруги, що відповідає потужності сигналу

## 2.4 Розробка корпусу пристрою-турелі

Розробка пристрою досить не проста задача, під час якої необхідно врахувати вплив одних компонентів на роботу інших таких як магнітні поля мотора можуть створювати завади для показань компасу, передача живлення між двома частинами, що обертаються одна відносно іншої чи такий банальний момент як радіочастотна прозорість пластику. Тре зазначити, що дана конструкція корпусу є конструктивним прототипом і потребує додаткової доробки та захисту від впливу зовнішніх чинників.

Згідно дослідження 3D друк в електромагнітній передачі [64] де проводилось ґрунтовне дослідження впливу матеріалу пластику на властивості антен, та здатність різними пластиками пропускати радіочастотне випромінювання, найкращим в плані проникності є ABS пластики з якомога меншою кількістю домішок таких як фарбники, тобто білі і сірі ABS пластики мають найбільшу радіочастотну прозорість, трішки  гірші характеристики проникності має сімейство PET пластиків, і абсолютно неприйнятним до використання є PLA. Оскільки наш пристрій все таки буде мати досить складну будову, з наявністю невеличких деталей ABS пластики у такому випадку можуть не забезпечувати надійної міцності. Тому краще використовувати більш міцні PET пластики, які мають ще й на додачу гнучку структуру, що гарантує відновлення деталей, після їх незначної деформації внаслідок фізичного впливу.

Одними з найважчих деталей турелі є елементи живлення, тому доцільно їх розміщувати на стаціонарній платформі, щоб не втрачати енергію на їх обертання, і логічно розмістити на ній же обв’язку для контролю заряду та перетворення напруги, що дозволить до-заряджання пристрою під час роботи. Також нижня платформа слугує кріпленням, для штифта-осі, на якому закріплені контактні підшипники рисунок 2.14 для обертальної частини, та інші статичні елементи.

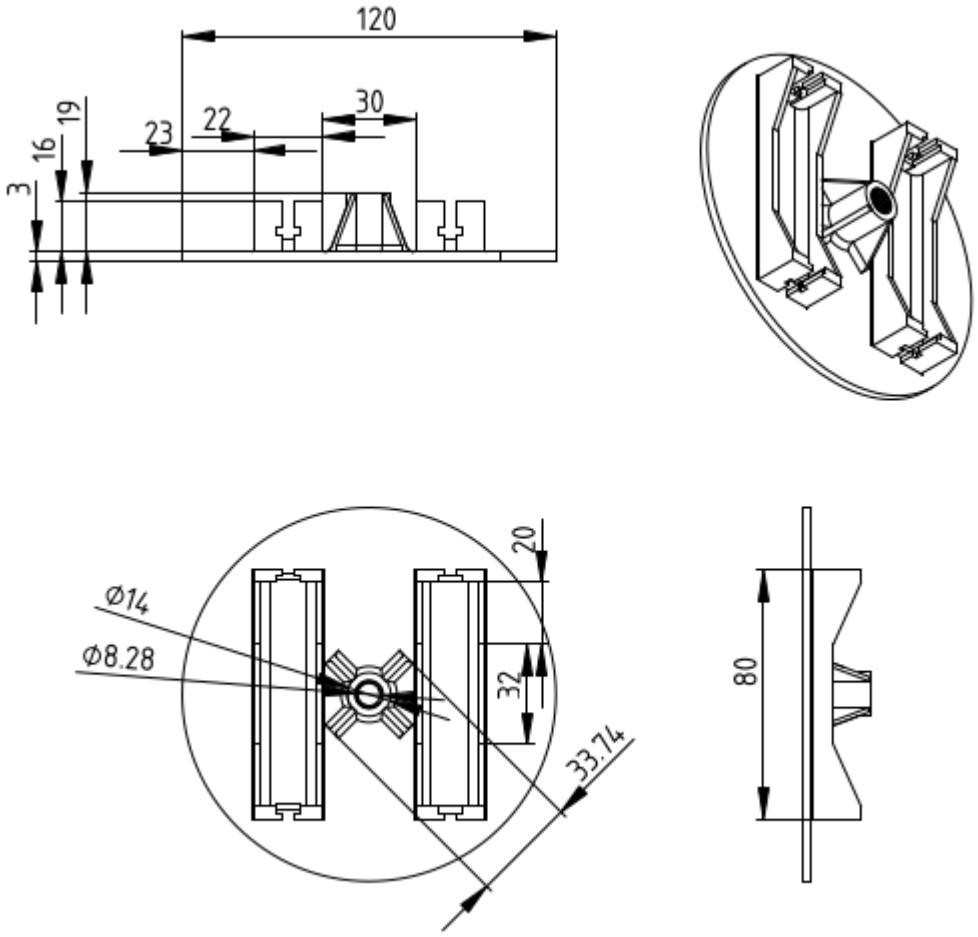



Рисунок 2.14 — Нижня платформа турелі

Платформа має отвір з різьбою куди монтується пін-вісь. Пін вісь повинен друкуватись з двох частин горизонтально, що збільшує його міцність за рахунок відсутності того, що шари пластику накладаються горизонтально. Сам штифт являє собою, двосторонню шпильку, пусту всередині для прокладання дротів. Та з отворами для виводу зсередини цих дротів рисунок 2.15 та шестигранний контактний майданчик для розміщення шестерні, з стопором для контактів підшипників над ним.

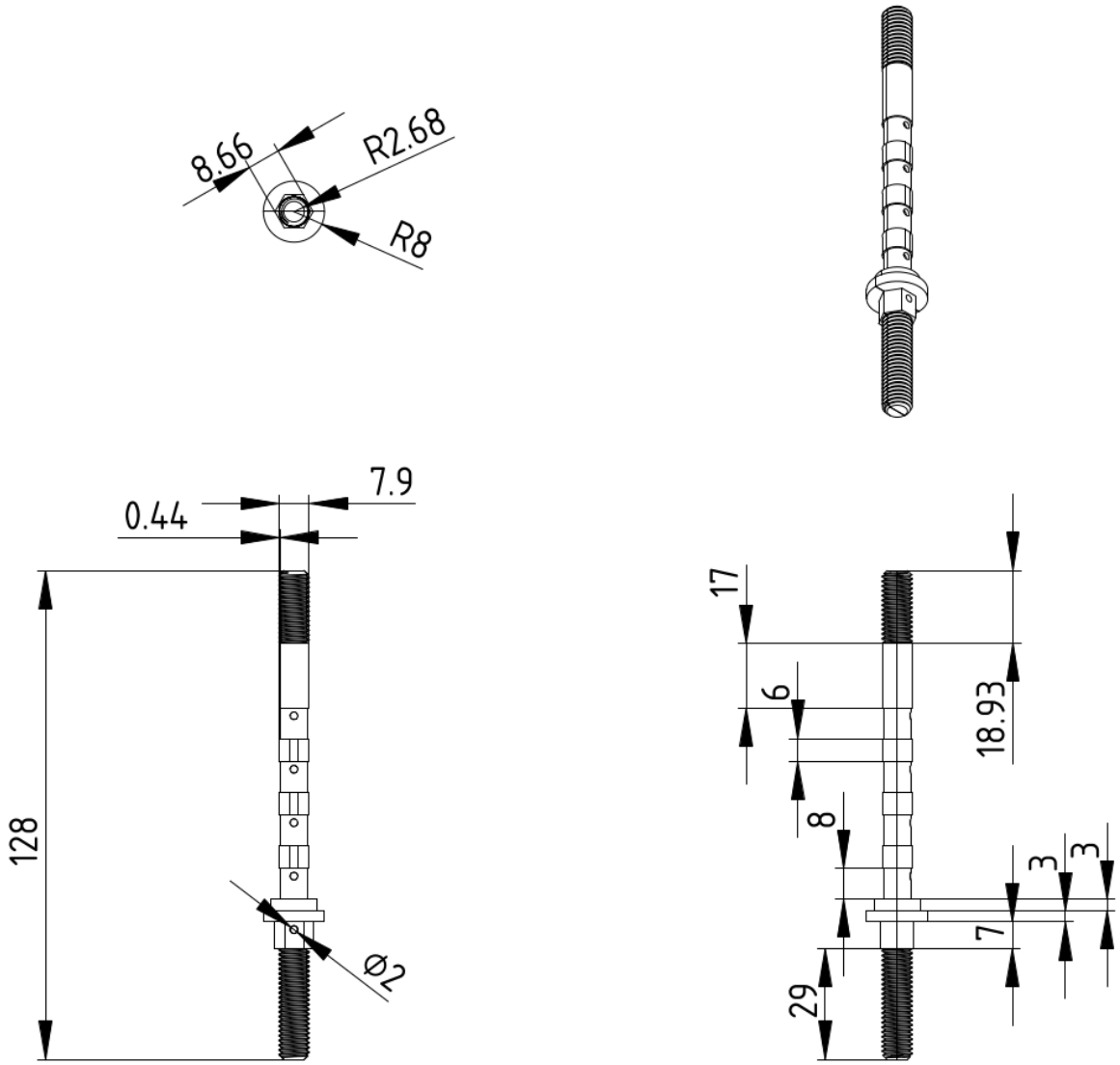



Рисунок 2.15 — Пін-вісь для кріплення контактних підшипників

Шестерня опори має монтажні отвори для прокладання дротів, що відповідають отворам для виведення дротів з віссю та шестигранний отвір парування з піном-віссю, що і служить опорою для обертання всієї турелі рисунок 2.16.

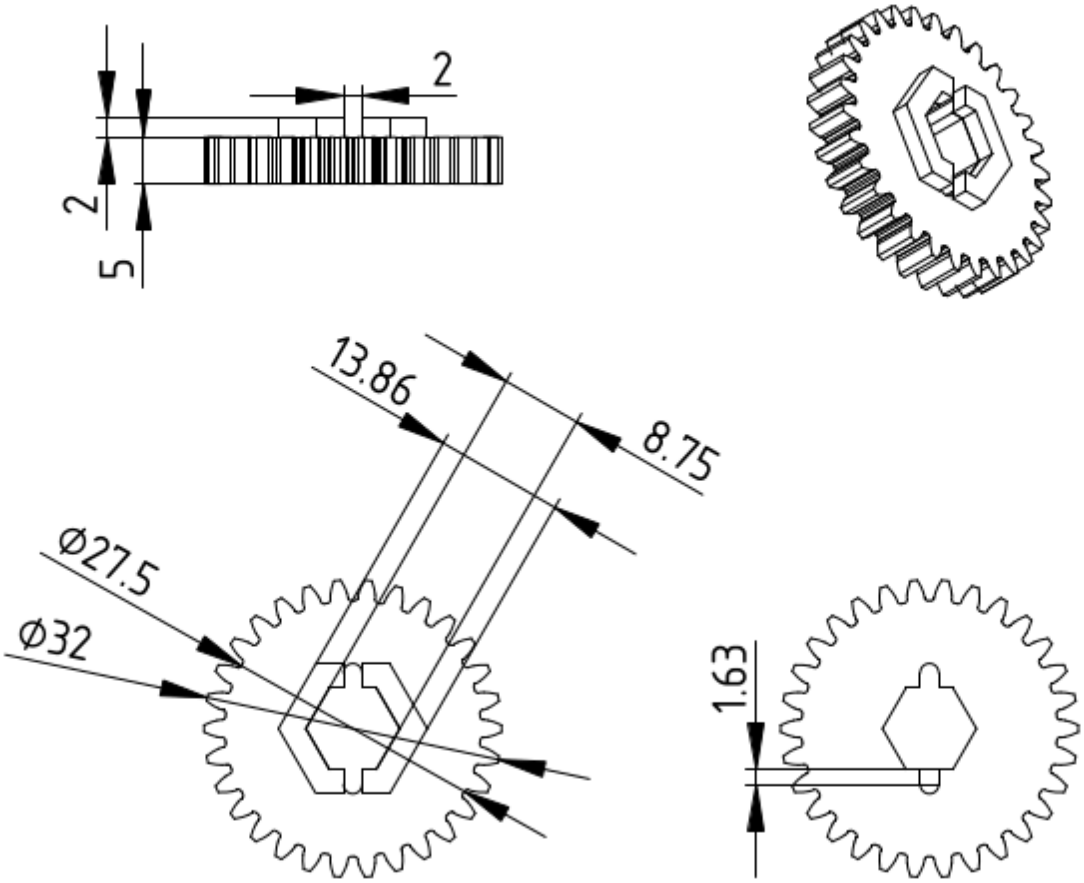



Рисунок 2.16 — Опорна шестерня на штифт вісь

Шестерня крокового двигуна за допомогою якої приводиться в рух поворотна частина турелі, відповідає розмірам опорної шестерні, тобто передаточне число 1:1,  і має отвір для з’єднування з кроковим двигуном рисунок 2.17.

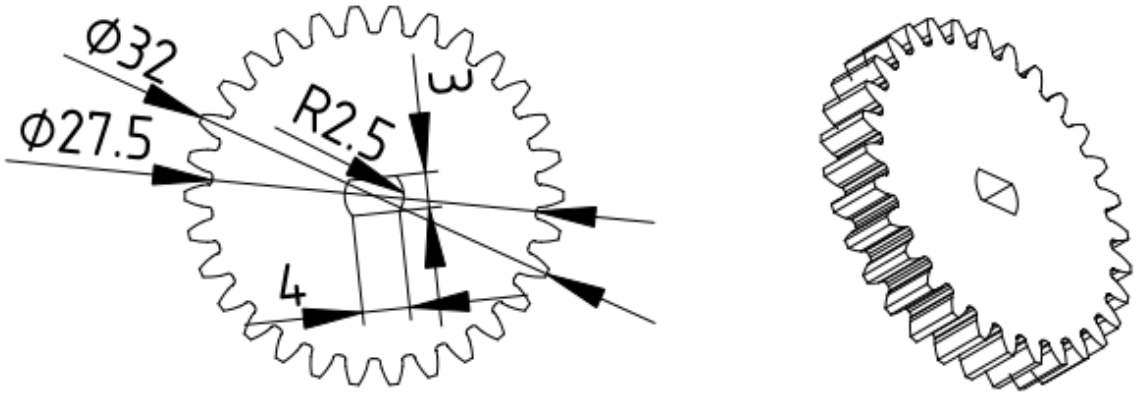



Рисунок 2.17 — Шестерня крокового двигуна

На штифт-осі розміщаються контактні підшипники 608zz, кожен підшипник поміщається всередині тримача, що має отвір для заведення дроту, зовні та всередині використовуються нікельовані пластини для пайки дротів, контакт між пластинами і підшипниками забезпечується методом прес-фітингу, між собою підшипники розділені циліндричними спейсерами.

Нижній тримач підшипника інтегрований в платформу, на платформі в свою чергу розміщується мікроконтролер, крововий двигун та його драйвер. Також на платформі знаходиться нижня точка монтування вимірювальної антени, а з протилежної сторони розташований канал-плече для винесення LoRa модулю та компасу, для розподілення маси в противагу антені, тре зауважити, що поряд з каналом з тією ж метою знаходиться отвір, куди монтується крововий двигун.

Верхній тримач підшипника зінтегрований з верхньою точкою монтування вимірювальної антени та розпіркою для з’єднання з каналом шлейфу LoRa/compass це забезпечить збільшення міцності конструкції. Збірна модель поворотної частини турелі зображена на рисунку 2.18.

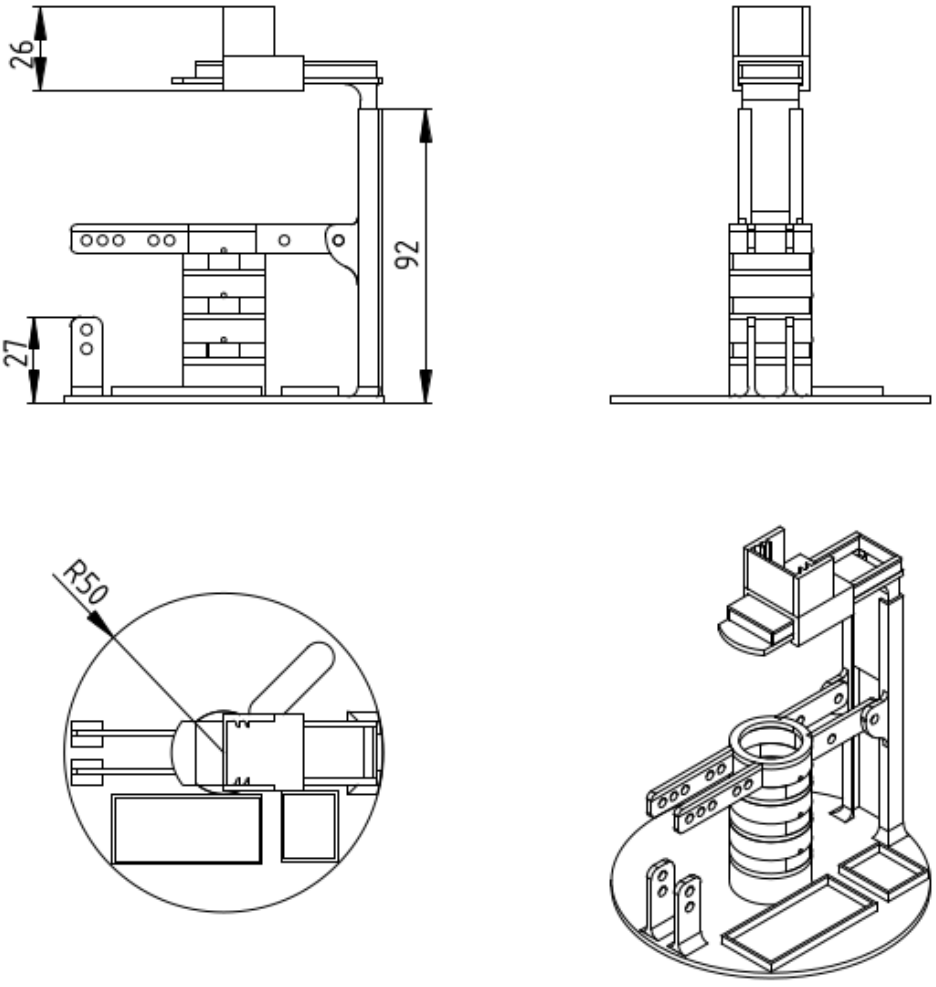



Рисунок 2.18 — Збірна модель поворотної частини турелі

Пристрій містить GPS модуль, якому дуже бажане постійне стабільне положення в просторі і відсутність обертання, особливо для початкової фіксації, а також наявність відкритого простору над ним. Тому GPS модуль розташовується на верхній платформі, яка водночас виконує функцію фіксатора рухомої частини на штифті, за рахунок наявності різьби, поряд передбачене місце для антени GPS рисунок 2.19. Потрібно зауважити, що підшипникові з’єднання мають дрижання контактів, тому на час зчитування даних з GPS турель бажано призупиняти, це не повинно викликати додаткової обробки, оскільки турелі і так необхідна зупинка для забезпечення синхронізації зчитування потужності радіовипромінювання та показників компаса.

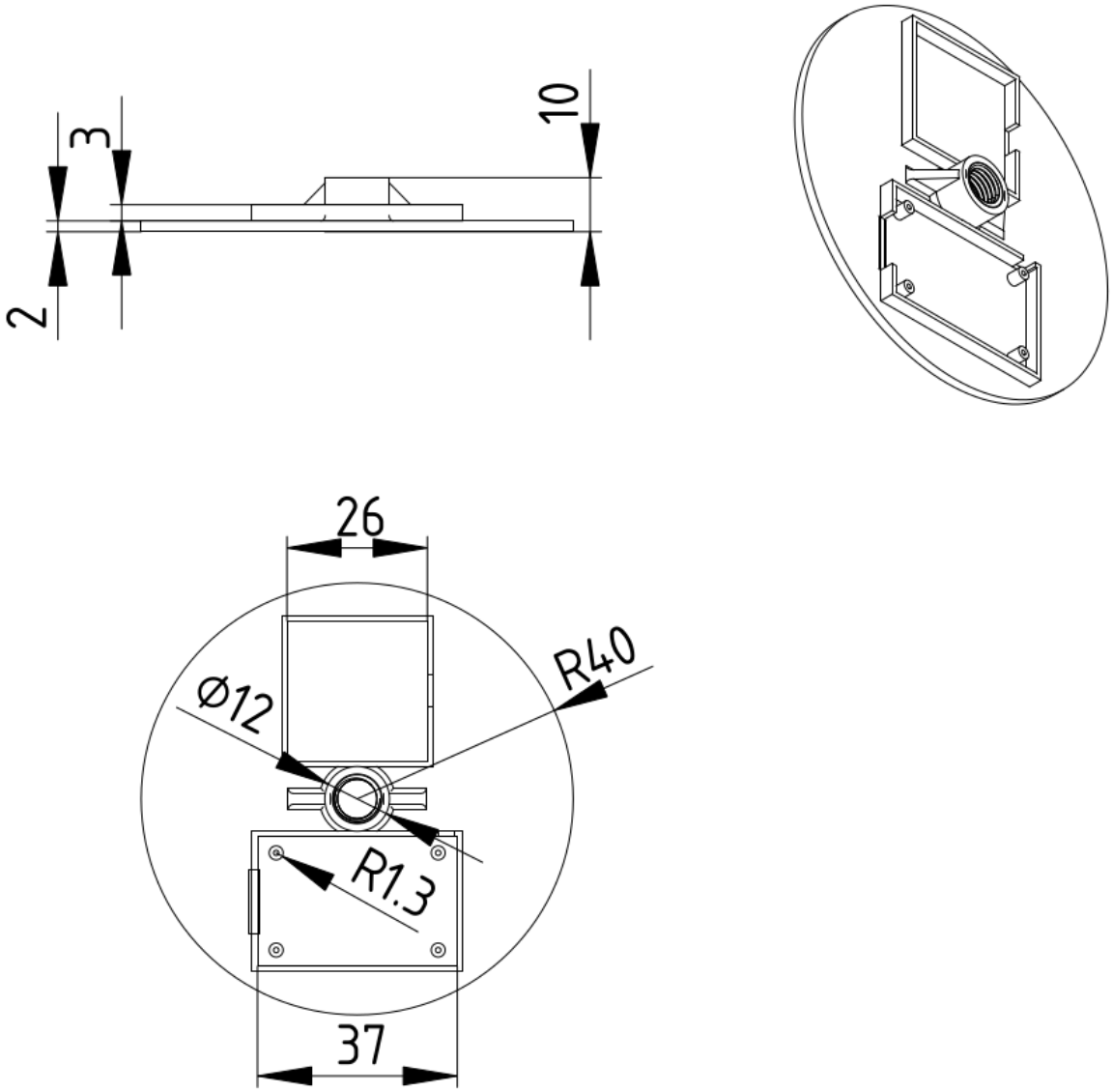
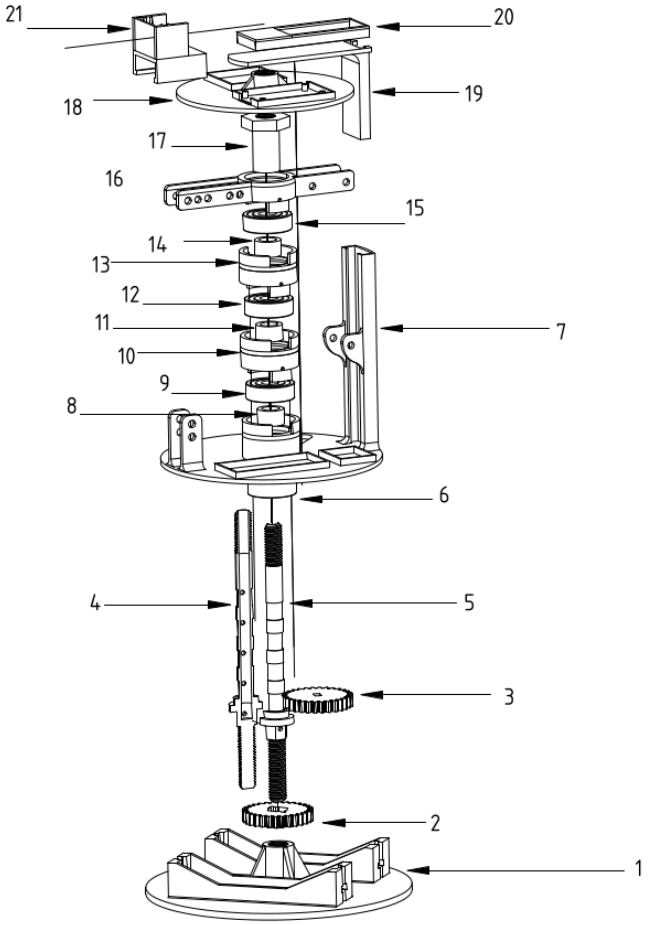



Рисунок 2.19 — Платформа для розміщення модуля та антени GPS

Компас знаходиться нагорі рухомої частини пристрою, якнайдалі від крокового двигуна, на спеціальному плечі, на цьому ж плечі розташовується модуль LoRa, це забезпечує йому розташування в найвищій точці пристрою, монтування на рухомій частині не впливає на якість передачі, оскільки в модулі використовується омнідирекційна антена.

Вимірювальна антена кріпиться в двох точках, на капронових стійках для максимального зменшення впливу корпусу на характеристики антени.

Пристрій турель складається з 21-го механічного компонента, розібрана система показана на рисунку 2.20.




| | |
|---|---|
| 1 | Bottom platform |
| 2 | Rod gear |
| 3 | Mottor gear |
| 4 | Inner rod part with holes |
| 5 | Inner rod no holes side |
| 6 | Bearing 608zz |
| 7 | Rotatable platform |
| 8 | Bearing spacer |
| 9 | Bearing 608zz |
| 10 | Bearing enclosure |
| 11 | Bearing spacer |
| 12 | Bearing 608zz |
| 13 | Bearing enclosure |
| 14 | Bearing spacer |
| 15 | Bearing 608zz |
| 16 | Top bearing enclosure |
| 17 | Lock nut |
| 18 | Top platform |
| 19 | Knee wire guide |
| 20 | Compass holder |
| 21 | LoRa holder |

Part List

Рисунок 2.20 — Розбірна схема турелі та перелік компонентів корпусу

Прототип пристрою в зборі зображений на рисунку 2.21.

Рисунок 2.21 — Прототип пристрою турелі

В додатку Б наведені креслення корпусу пристрою турелі.

## 2.5 Схема електрична принципова пристрою-приймача

Пристрій приймач являє собою значно спрощену версію попередньо описаної турелі та виконує функцію пристрою-проксі. Основна мета якого отримати дані по інтерфейсу LoRa, виконати попередню обробку цих даних та передати ці дані на комп’ютер.

Оскільки в даному пристрої системи LoRa U10 використовується лише для приймання даних, то відповідно і струм споживання перебуває в межах 60-70mA, що цілком може забезпечити модуль Arduino. Тому для спрощення схеми доцільно використовувати джерело напруги 3,3V з модуля Arduino. Також

живлення пристрою здійснюється за допомогою USB порту комп’ютера. Загальний вигляд принципової схеми зображений на рисунку 2.22.

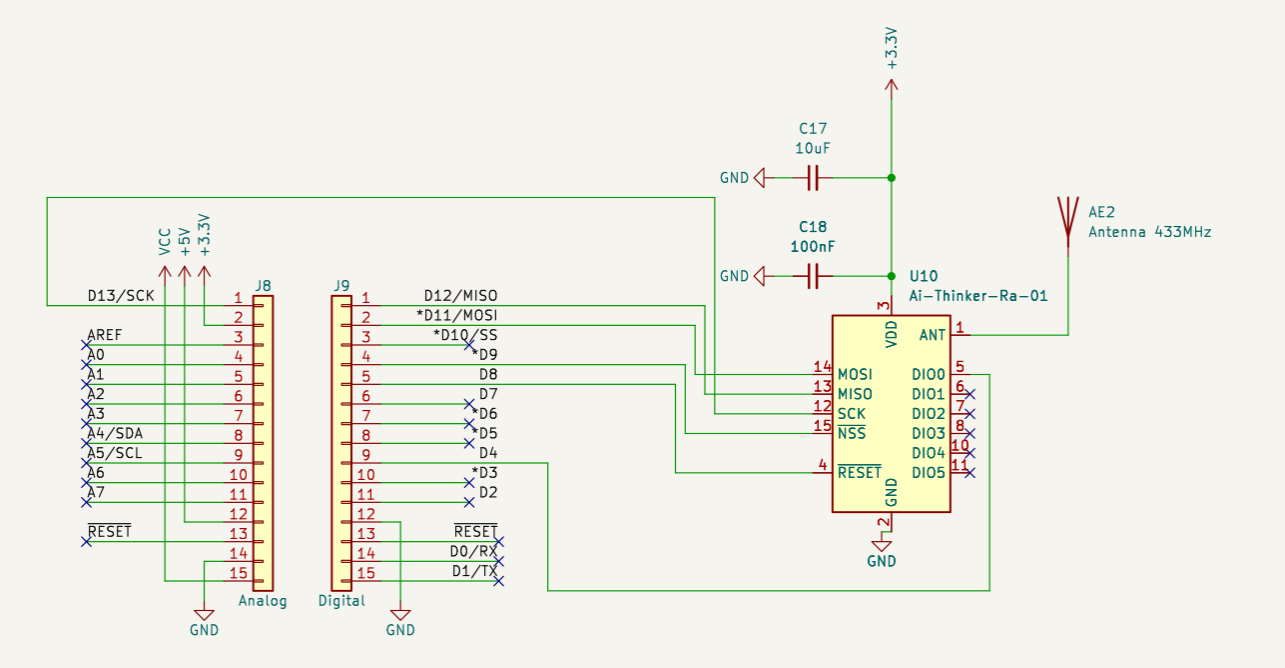



Рисунок 2.22 — Пристрій приймач LoRa

Підключення модуля модуля LoRa Ra-01 U10 повністю аналогічне до того що виконано в турелі. Комунікація мікроконтролеру та модуля відбувається за допомогою SPI інтерфейсу, і так само виходи модуля MOSI, MISO, SCK, приєднанні до відповідних виходів мікроконтролера D11, D12, D13. NSS вихід модуля до D9 модуля мікроконтролера, Reset до D8, DIO0 до D4. Як вже було зауважено різниця лише в тому, що живлення LoRa отримує напряму з перетворювача на модулі Arduino.

В додатку Б наведена схема електрична принципова пристрою приймача.

## 2.6 Розробка алгоритму для роботи пристрою турелі

Пристрій турель є основною компонентою системи вимірювання. Оскільки пристрій побудований на мікроконтролері, і його задачею є збір показників датчиків, то програма має виконуватись в нескінченному циклі і єдиною умовою виходу з неї є припинення подачі живлення. Налаштування датчиків та параметрів мікроконтролера в пресетап блоці, ініціалізація всіх інших параметрів в Setup модулі.

В головному циклі програми посилаються команди крокoвому двигуну для повороту турелі на визначену кількість кроків, турель зупиняється, для того щоб усунути можливі дрижання контактів під час передачі через контакти підшипники GPS даних, а також щоб гарантувати відповідність зчитаних даних компаса до зчитаних даних потужності випромінювання.

Наступним етапом починається зчитування показників датчиків тобто з компасу та даних потужності радіочастотного випромінювання з логарифмічного перетворювача, тут же відбувається перетворення значень напруги в децибел на міліват. Для економії процесорного часу, у випадку якщо використовується програмний серійний порт, цей захід є навіть необхідним, зчитування даних GPS відбувається лише кожних 60 секунд, за умови що вони доступні, тобто GPS датчик отримав фікс і в серійному порті присутні дані.

З отриманих даних давачів формуються повідомлення в LoRa буфер, до буферу застосовується алгоритм шифрування. І далі зашифрований вміст буферу транслюється за допомогою LoRa протоколу в радіоефір. Загальна схема алгоритму роботи пристрою турелі зображена на рисунку 2.23.

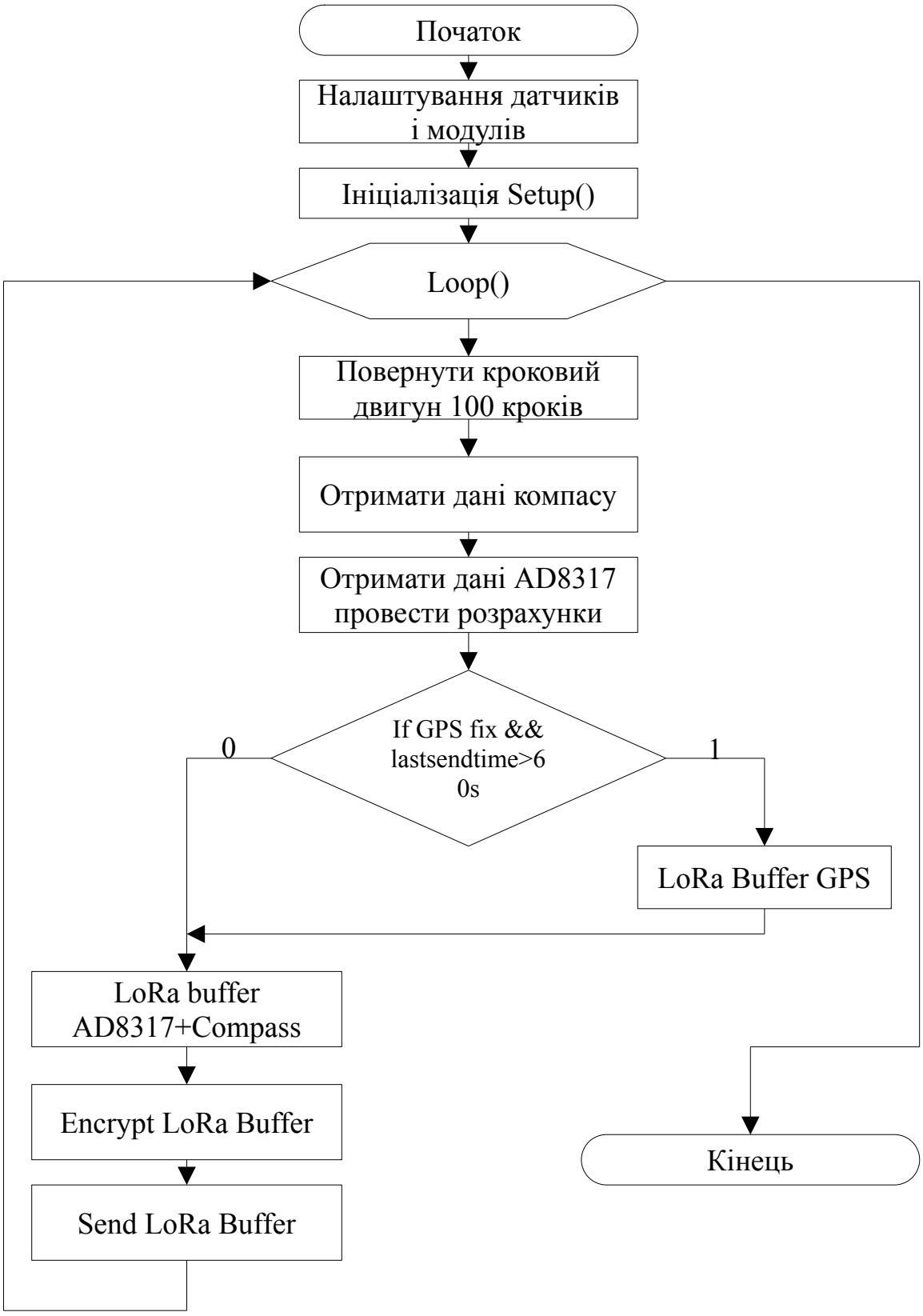



Рисунок 2.23 — Схема алгоритму роботи пристрою турелі

## 2.7 Розробка алгоритму для роботи пристрою приймача

Пристрій приймач виконує роль ретранслятора протоколу LoRa на персональний комп’ютер. Аналогічно умовою припинення роботи пристрою є зникнення живлення.

Основна програма ініціалізує модуль LoRa  в режимі читання, як тільки дані надходять по протоколу LoRa, буфер з даними розшифровується і дані передаються на ПК. Загальна схема алгоритму роботи пристрою приймача зображена на рисунку 2.24.

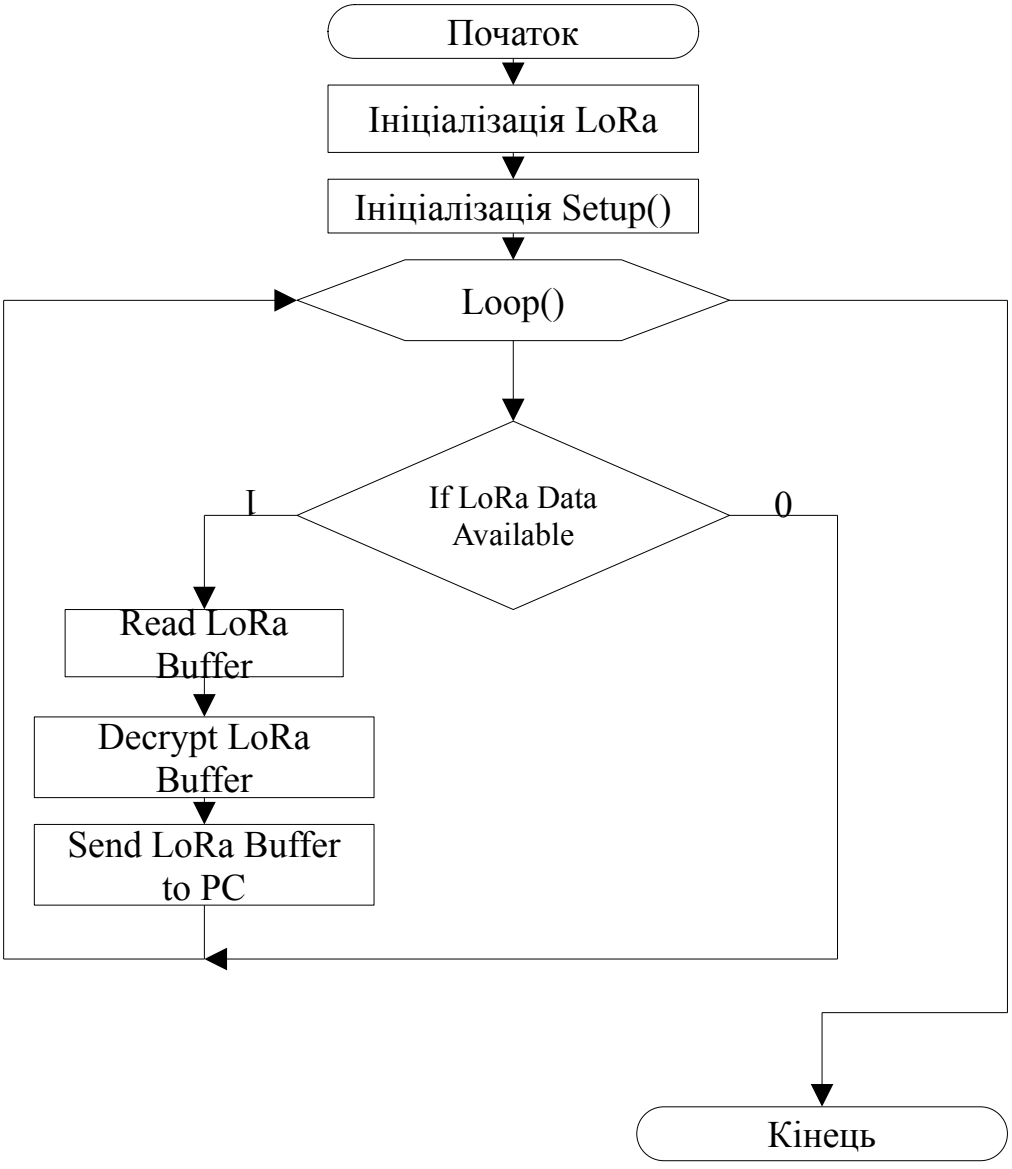



Рисунок 2.24 — Схема алгоритму роботи пристрою приймача

## 2.8 Розробка алгоритму функціонування системи

Для того алгоритмізувати систему визначення джерела живлення ми проробили велику роботу, щоб структурувати компоненти такої системи.

Отже одним з компонентів системи є пристрій для вимірювання даних, попереднього перетворення їх, формування пакетів, по спеціально розробленому для цього протоколу та передачі цих пакетів в зашифрованому вигляді на проміжний пристрій приймач.

Пристрій приймач повинен правильно ідентифікувати приналежність пакету саме йому, декриптувати вхідне повідомлення, провести попередню обробку прийнятого повідомлення і відправити його на відображення на дисплей монітору комп’ютера.

Останній компонент системи це програма, що здатна відобразити прийняті данні на моніторі комп’ютера. Та вести історію, прийнятих значень, на основі яких можна будувати різноманітні візуалізації.

Всі компоненти системи є важливими і вся система в цілому не може обійтись без жодного з трьох компонентів. Оскільки в системі наявні три блоки, кожен має керуватись своїм програмним засобом.

В додатку Б наведено алгоритм функціонування системи визначення джерела радіовипромінювання.

# 3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

В розділі проведено огляд готових рішень, що існують для реалізації компонентів системи, проведено їх аналіз здійснено вибір кращих методів їх реалізації для нашого випадку і використання в нашій системі.

## 3.1 Розробка програмного забезпечення для пристрою турелі

Отже мовою ґрунтуючись на розділі 1 мовою програмування програмування для реалізації програми для контролю пристрою турелі було обрано C++ на базі фреймворку Arduino IDE. Це дозволяє використати нам готові рішення по роботі з модулями та реалізації протоколів, як вбудованих, так і вільно доступних модулів.

Нам знадобиться бібліотека, що дозволяє комунікувати з серійними SPI інтерфейсами *#include <SPI.h>* ця бібліотека є вбудована і не потребує додаткового встановлення.

Також нам потрібна бібліотека для роботи з I2C інтерфейсом *#include <Wire.h>* ця бібліотека дозволяє комунікувати з I2C/TWI пристроями та на Arduino R3 layout які ми використовуємо є визначені виходи SDA (data line) та SCL (clock line) відповідно A4 (SDA), A5 (SCL).

Також додатково нам знадобиться бібліотека віртуального серійного порту UART для спілкування з GPS модулем *#include <SoftwareSerial.h>* незважаючи на недоліки програмного серійного порту, а саме не оптимізоване використання процесорного часу, це все таки дозволяє використовувати найпростіші Arduino плати, в подальшому можливе вдосконалення програми для підтримки і апаратного UART.

Оскільки в програмі необхідна робота з компасом ми використаємо бібліотеку *#include <QMC5883LCompass.h>* , яку необхідно додатково додати до Arduino IDE знайти можна по імені QMC5883LCompass авторства mprograms. Ця бібліотека на відміну від схожих реалізацій роботи з даним з чіпом 5883 дозволяє проводити калібрування компасу.

Для роботи з NMEA протоколом, використаємо одну з найлегших в плані використання пам’яті бібліотеку *#include <TinyGPS++.h>* яку можна встановити додавши пакет TinyGPSPlus від mikalhart в менеджері бібліотек Arduino IDE.

Для роботи з LoRa модулем нам знадобиться бібліотека *#include <LoRa.h>*, яку теж можна встановити за допомогою менеджера бібліотек знайшовши її за іменем arduino-LoRa та автором sandeepmistry. Це одна з найменших, за вимогами пам’яті бібліотек для роботи з LoRa.

Оскільки робота пристрою та обертання мають бути синхронізовані для роботи з кроковим двигуном використаємо бібліотеку #include <Stepper.h> для синхронної роботи з двигунами. Бібліотека також має бути встановлена за допомогою менеджера бібліотек, знайти її можна за іменем бібліотеки MobaTools авторства MicroBahner.

Для реалізації шифрування нам знадобляться дві бібліотеки #include <Crypto.h> #include <AES.h>. Що поставляються універсальним пакетом шифрування crypto від автора rweather. Бібліотека має гарний аналіз алгоритмів шифрування, з порівняльною таблицею часу їх виконання [65]. На основі цієї таблиці можна зробити висновок, що основним критерієм для нас все таки є час виконання шифрування оскільки надійність алгоритму значно сповільнює роботу турелі, тому використаємо AESTiny128, який на даний момент вже не вважається надійним алгоритмом шифрування. Збільшення надійності може компенсуватись частішою зміною ключів. AESTiny128 має менший слід в пам’яті порівняно AES128, але підтримує лише шифрування і не має

можливості дешифрувати, тобто в нашому випадку оскільки пристрій використовується лише для передавання даних, він чудово підходить. В подальших вдосконаленнях системи можна з вимогами двостороннього спілкування необхідно буде використовувати AES128. Середній час пакування на шаблонних даних на Arduino Nano з частотою ядра 16MHz приблизно біля 50us.

Встановлюємо ініціалізаційні дані для роботи крокового двигуна

#define STEPS 100: Ця директива препроцесора визначає константу STEPS і призначена для визначення загальної кількості кроків, яку повинен виконати кроковий двигун для одного циклу обертання. У цьому випадку вказано 100 кроків. Що забезпечує достатню швидкість обертання, з достатнім моментом.

Stepper stepper(STEPS, 5, 7, 6, 10) створює екземпляр об'єкта класу Stepper з параметрами ініціалізації. Параметри включають: STEPS - Кількість кроків на що має виконати двигун (попередньо встановлено за допомогою #define). 5, 7, 6, 10 - піни мікроконтролера, до яких підключені витки крокового двигуна. Порядок передачі цих значень важливий і визначає правильну послідовність включення котушок для забезпечення правильної послідовності кроків.

Структура RFDATA, призначена для зберігання даних, отриманих від логарифмічного перетворювача.

Структура RFDATA містить такі поля:

- avg_rawADC, max_rawADC, min_rawADC: Ці поля відображають середнє, максимальне та мінімальне значення, отримане від аналого-цифрового конвертора (ADC);
- min_V, max_V, avg_V: Ці поля вказують на мінімальне, максимальне та середнє значення напруги сигналу;
- min_dbm, max_dbm, avg_dbm: Ці поля вказують на мінімальну, максимальну та середню потужність сигналу у децибелах міліват (dBm).

Структура COMPASSDATA, яку ви навели, призначена для зберігання даних, отриманих від магнітометра та компасу. Давайте розглянемо кожне поле цієї структури:

- x, y, z: Ці поля відображають компоненти магнітного поля у трьох основних осях;
- a, b: Ці поля призначені для збереження значень азимуту та пеленгу (bearing);
- compass_direction: Це масив символів, який може містити напрямок, вказаний компасом. Розмір масиву обмежено трьома символами ("N", "S", "E", "W", "NW","SSW" і т.д.).

Для зберігання даних створені екземпляри структур adc8317_0 та compassdata, тепер вони можуть використовуватися для зберігання та доступу до даних.

Для роботи з LoRa в даній системі імплементовано дворівневий протокол, який буде описано в окремому розділі. Оскільки LoRa потребує досить не мало пам’яті, як мінімум на один пакет, нам потрібно створити глобальний буфер масив, з яким ми будемо працювати впродовж роботи програми, це нівелює проблему фрагментації пам’яті. LoRa пакет має обмеження в 256 байт, так як в нашому протоколі будуть використовуватись деякі байти як службові ми обмежимо розмір пакету встановивши LORA_BUFFER_LIMIT 210 байт і проініціалізувавши відповідний буфер char lora_buffer[LORA_BUFFER_LIMIT] заповнивши його нуль символами, що означає що це пустий рядок.

Для ідентифікації наших пакетів в LoRa проініціалізувати змінні байти ідентифікатори системи lora_trgtIdA=0x0B, lora_trgtIdB=0x80 по цьому ідентифікатору кожен пристрій системи буде знати, що це пакет саме нашої системи. Та lora_devIdA=0xAA і lora_devIdB=0xAA для ідентифікації конкретного пристрою, тобто комбінації цих байтів мають бути унікальні.

Оскільки в LoRa існує обмеження на розмір пакету, то необхідно передбачити можливість ідентифікації пакетів та груп пакетів, lora_msgCnt та lora_batchId відповідно, до яких вони відносяться ці ідентифікатори і будуть передаватись на нижньому рівні протоколу.

На нижньому рівні протоколу використовується псевдо JSON, для зменшення кількості байтів що необхідно передати, для цього створено кілька директив GPS_BASE_DATA_JSON темплейт для базових даних NMEA протоколу, що міститься в кожному апдейті. А далі змінні що розміщуються в EEPROM мікроконтролера для збереження пам’яті, для кожного типу GPS повідомлення, у тому ж форматі псевдо JSON:

GPS_SAT_DATA_JSON - для даних GPS щодо супутників;

GPS_COURSE_DATA_JSON - для даних курсу GPS;

GPS_DATE_DATA_JSON - для даних дати GPS;

GPS_TIME_DATA_JSON - для даних часу GPS;

GPS_HDOP_DATA_JSON - для даних точності HDOP GPS;

GPS_SPEED_DATA_JSON - для даних швидкості GPS;

GPS_LOCATION_DATA_JSON - для даних місцеперебування GPS;

GPS_ALTITUDE_DATA_JSON - для даних висоти GPS.

А також правила форматування даних магнітометра і виміряної потужності COMPASS_RF_DATA_JSON.

Даний підхід дозволяє значно спростити подальшу роботу з повідомленнями і формування пакетів, при цьому не використовуючи багато пам’яті, хоч і з першого погляду може здатись досить заплутаним, особливо завдяки тому, що sprintf_P функція Arduino дуже обмежена при роботі з деякими типами даних.

Для шифрування задамо тестовий ключ key, та створимо cipher_buffer[16] в якому будуть розміщатись дані, що шифруються на даний момент і проініціалізуємо екземпляр класу AESTiny128 aestiny128.

Для вимірювання і обробки сигналу від логарифмічного перетворювача AD8317, який вимірю потужність RF-сигналу, необхідно визначити ряд параметрів:

AD_RESOLUTION — роздільна здатність аналого-цифрового перетворювача;

aref_voltage — константа, яка визначає напругу на ref піні для аналого-цифрового перетворювача. У даному випадку, встановлено значення 2.048V;

AD8317_INPUT — визначає аналоговий пін Arduino куди підключений AD8317, у даному випадку це вихід A0;

NUM_SAMPLES - константа, яка визначає кількість вибірок, які беруться для обчислення середнього значення та вибору максимального серед них, у даному випадку це 20 раз;

sample_count, min, max, sum — вказує на активне замірювання на даний момент часу;

dbm_at_0V — каліброване значення показника напруги, виміряне за допомогою TinySA на частоті 2,4GHz тобто саме на цих частотах рівень сигналу буде найточнішим.

MV_DB_SLOPE - похила лінійної функції, що описує характер зміни напруги в залежності від частоти і визначається методом включенням AD8317 в схему, у нашому випадку це 0,022.

Далі програма визначає піни програмного серійного порту RXPin = 2 TXPin = 3 та створює відповідний екземпляр класу gpsSerial для роботи з GPS датчика.

Також створюється екземпляр класу compass для роботи з компасом.

Останнім сконфігуруємо піни LoRa ss до 9, rst до 8, dio0 до 4 з якими вона і буде проініціалізована.

Класична програма на Arduino містить функцію setup() де відбувається початкова ініціалізація пристроїв, з попередньо визначеними даними, що ми розглянули раніше. Ця функція викликається один раз на початку програми, відповідно тут знаходяться лише ті блоки програми які необхідно викликати на початку. Тут ми встановлюємо ключ шифрування, встановлюємо зовнішню опорну напругу, швидкість серійного порту, ініціалізуємо компас, та встановлюємо швидкість порту спілкування з модулем GPS. Необхідно звернути увагу, що компас ініціалізується compass.setSmoothing(3, true) значення 3 підібране експериментальним способом, і визначає наскільки стабільний будуть показники компаса, тобто це усереднення яке відбувається всередині бібліотеки. Також оскільки в нашому випадку використовується LoRa модуль частотою 433MHz відповідний має і вигляд функція ініціалізації LoRa.begin(433E6).

Функція loop() викликається безперервно після виконання функції setup(). Розглянемо її структуру та дії, які вона виконує:

- stepper.step(100) - дає крововому двигуну команду повернутись на 100 кроків, що приблизно дорівнює 15 градусів (повне коло обертання відбувається за 25-30 секунд);
- ReadCompass() - функція, що вичитує дані з компасу в структуру compassdata;
- ReadAD8317() - виклик функції, що збирає заміри з AD8317, робить підрахунки та записує результати в структуру adc8317_0;

- Формування псевдо JSON пакетів, за темплейтом COMPASS_RF_DATA_JSON, з допомогою функції sprintf_P результат зберігається у буфері lora_buffer;
- LoraSend() - виклик функції, що відправляє по LoRa дані, що знаходяться в lora_buffer;
- GPSToSerial() викликається лише тоді коли дані GPS є доступні. Виклик LoraSend() відбувається всередині для кожного типу GPS повідомлень;
- Останнім кроком збільшується ідентифікатор групи пакетів lora_batchId, що і сигналізує закінчення одного циклу вимірювання.

Зупинимось детальніше на основній функції програми, а саме ReadAD8317(), вона призначена для зчитування та обробки даних з аналого-цифрового перетворювача (ADC), який вимірює сигнал від пристрою AD8317, що використовується для вимірювання потужності радіочастотного сигналу. Давайте розглянемо її структуру та кроки, які вона виконує:

- Ініціалізація тимчасових змінних: ADCReading, sample_count, min, max;
- тимчасові змінні оновлюються з кожним циклом вимірювання;
- після закінчення циклу три змінних max_V, min_V, avg_V, структури adc8317_0;
- Для кожного результату вимірів напруги викликається функція convertVoltageToDbm, що перетворює значення напруги в значення dBm dbm_at_0V каліброваного значення при нульовій напрузі величину, MV_DB_SLOPE коефіцієнт нахилу функції функція описується формулою 3.1.

$$dBm = dbmat\,0\,V - (voltage / MVDBSLOPE) \tag{3.1}$$

Шифрування пакетів LoRa виконується перед кожною відправкою пакету, шляхом викликання функції int encryptLoraBuffer(char* lora_buffer,char* buffer).

Реалізація цієї функції спрямована на використання якомога менше пам’яті, лише не значно більше 16 байт, і виконує поступове шифрування блоків даних, збережених у буфері lora_buffer, за допомогою алгоритму AES (Advanced Encryption Standard) в режимі ECB (Electronic Codebook). Розглянемо її загальний опис:

Функція має два вхідних параметри lora_buffer вхідний буфер, який містить дані для шифрування, та невеличкий 16 байтний buffer буфер для зберігання шифрованих даних.

Для шифрування блоків функція використовує цикл while, який працює, доки не вичерпається вхідний буфер. Кожен блок (16 байт) даних з кінця lora_buffer копіюється в buffer. Блок даних шифрується алгоритмом AES в режимі ECB. Шифровані дані заміщують оригінальні дані в lora_buffer. Обчислюється час виконання для оцінки продуктивності шифрування

Ця функція служить для захисту конфіденційної інформації, такої як дані передачі LoRa, використовуючи алгоритм шифрування AES.

В додатку В наведено повний лістинг програми для пристрою турелі.

## 3.2 Розробка програмного забезпечення для пристрою приймача

Основним призначенням цього пристрою є отримання LoRa-пакетів і трансляція отриманих пакетів по серійному протоколу програмному засобу запущеному на операційній системі.

Давайте розглянемо кожну частину програми:

- Включення бібліотек та визначення констант в цілому даний блок є спрощеним варіантом пункту 3.1 за виключенням відсутності давачів.

- Функція setup ініціалізує модуль AES з встановленим ключем. Зауважте ключі на всіх пристроях мають бути ідентичні. На відміну від програмного забезпечення турелі цей пристрій має дешифрувати дані тому тут ми не можемо використовувати tinyAES і ініціалізуємо повноцінний модуль алгоритму.
- Основний цикл loop чекає та обробляє нові пакети, що надходять через LoRa.
- Одна з основних функцій програми це decryptLoraBuffer
    - функція для розшифрування LoRa-пакету;
    - використовується AES у режимі ECB;
    - зчитує пакет блоками по 16 байт;
    - розшифровані дані заміщують оригінальні дані.

Цей код призначений для приймання і розшифровування LoRa-пакетів. Розшифровані дані та ідентифікатор пакету виводяться через Serial порт для подальшого аналізу або використання.

В додатку В наведено повний лістинг програми для пристрою приймача.

## 3.3 Опис протоколу для спілкування компонентів системи

Оскільки в системі присутні як мінімум три компоненти, з різних світів IT технологій, що поєднує вимоги до розміру переданих даних, економії пам’яті, частково пропрієтарних протоколів таких як LoRa і такого іншого. Це породжує вимоги до особливого протоколу спілкування компонентів системи.

Отже вимоги до такого протоколу:

- мінімальний слід в пам’яті при цьому без втрати функціональності;

- ідентифікація пристроїв системи;

- надійність проти перемішування вимірювань;

- легкість обробки даних,

- розширюваність.

Протокол спілкування пристроїв за описом можна розкласти на два рівні:

- низький рівень — саме рівень спілкування між LoRa пристроями, тобто це протокол надбудова над протоколом LoRa, саме цей рівень забезпечує ідентифікацію пристроїв систем;

- високий рівень — саме рівень, що описує в якому форматі посилаються дані але вони ніяк не впливають на логіку роботи роботи LoRa пристроїв.

Та все ж ці рівні знаходяться в одному пакеті, тому їх можна також класифікувати за функціоналом в одному повідомленні за критерієм: заголовок (header) та тіло (body).

Під заголовок відведено 6 перших байт LoRa пакету, під тіло весь залишок тобто 250 байт, що може містити як текстові так і бінарні дані.

Перевірка перших двох байт гарантує, що повідомлення відправлене саме нашою системою, випадковий збіг перших двох байт можливий але вкрай малоймовірний. Це дозволить швидко відфільтровувати непотрібні повідомлення від інших систем. В подальшому APPID (application ID) можна буде використовувати, для службових повідомлень, наприклад розробити систему динамічної роздачі ідентифікаторів пристроям.

Другий та третій байти є унікальним ідентифікатором пристрою DEVID, теоретично дає необмежену кількість пристроїв з унікальними ідентифікаторами 2^16 тобто 65536 варіацій, це перевищує будь-яке значення пристроїв в системі при здоровому глузді.

Четвертий байт ідентифікатор групи повідомлень (batch id), тобто в якомусь сенсі це спосіб сказати, що всі повідомлення, з такою групою, були відправлені разом і повинні розглядатись як одна група.

П’ятий байт є унікальний ідентифікатор повідомлення в межах групи (message id). В якомусь сенсі це дозволяє зберегти порядок, хоча і теоретично повідомлення не можуть мікшуватись.

Четвертий і п’ятий байт можуть змінюватись в межах від 20 до 256, щоб уникнути проблем з не друкованими символами. Такий підхід гарантує унікальність повідомлень та надають можливість аналізу проблем.

Демонстративно структуру розподілу байтів протоколу зображено на рисунку 3.1.

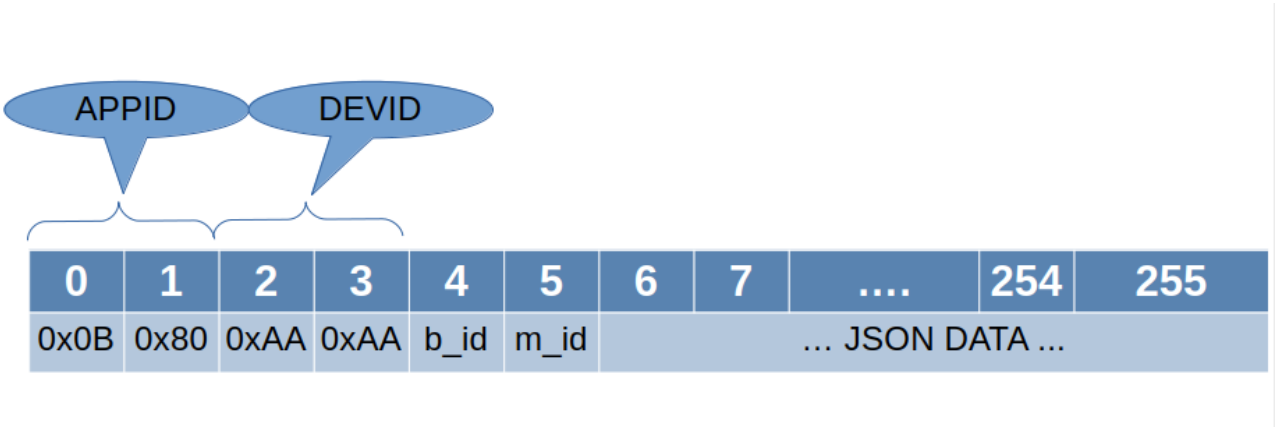



Рисунок 3.1 — Структура протоколу системи

Оскільки використання протоколу LoRa накладає суттєві обмеження розміру, то використання класичного JSON  не є доцільним, як формату з надлишковими даними. Але і його ж простота дозволяє значне спрощення формату, тобто це вже буде не валідний JSON, але його цілком можна називати

псевдо-JSON за його схожість. Тому саме для передачі даних по LoRa буде використовуватись цей псевдо-JSON.

Перша разюча відмінність це відсутність кореневої ідентифікації масив чи об’єкт. Кореневий елемент має просто ім’я. Так тут можлива подальша оптимізація, наприклад якщо елемент об’єкт чи масив то можливо видаляти двокрапку, але в нашому випадку на даний момент міститься лише один вкладений об’єкт і видалення лише одного символу не впливає на розмір, але додає більше логіки.

Друга особливість це, видалення подвійних квот навколо кожного імені елемента, тобто кількість зекономлених байтів це кількість елементів $n \times 2$. При цьому зберігається можливість додавати квоти навколо значень, для ідентифікації типу даних string.

Приклад такого повідомлення в системі, для компасу:

compass:{x:200,y:1000,z:500,a:30,b:15,d:"NNE"}

Використання такого формату, лише на прикладі такого короткого повідомлення, призводить до економії 16 байт порівняно з справжнім JSON форматом, а це приблизно 30% від розміру оригінального повідомлення, при цьому зберігається функціонал формату.

## 3.4 Розробка аплікації для прийому та обробки даних на ПК

Тут ми розглянемо розробку програмного забезпечення, що приймає дані з пристрою приймача під’єднаного до головної станції на персональному комп’ютері.

Оскільки програма розробляється на QT портування її на інші пристрої такі як смартфони чи планшети не має викликати особливих зусиль, і зазвичай лише полягає в встановлені необхідного фреймворку, для збірки.

Програма має одно-віконний інтерфейс, що містить дві таби:

- DataView де область розділена на текстове поле зліва, і відображає дані отримані з пристрою приймача, з перетвореним форматом з псевдо-JSON в звичайний JSON, в правій частині знаходиться лог таблиця з таймстемпом, та даними які візуально схожі на ліву текстову таблицю, але вони вже пройшли розпарсювання JSON класом, загальний вигляд таби DataView показаний на рисунку 3.2;

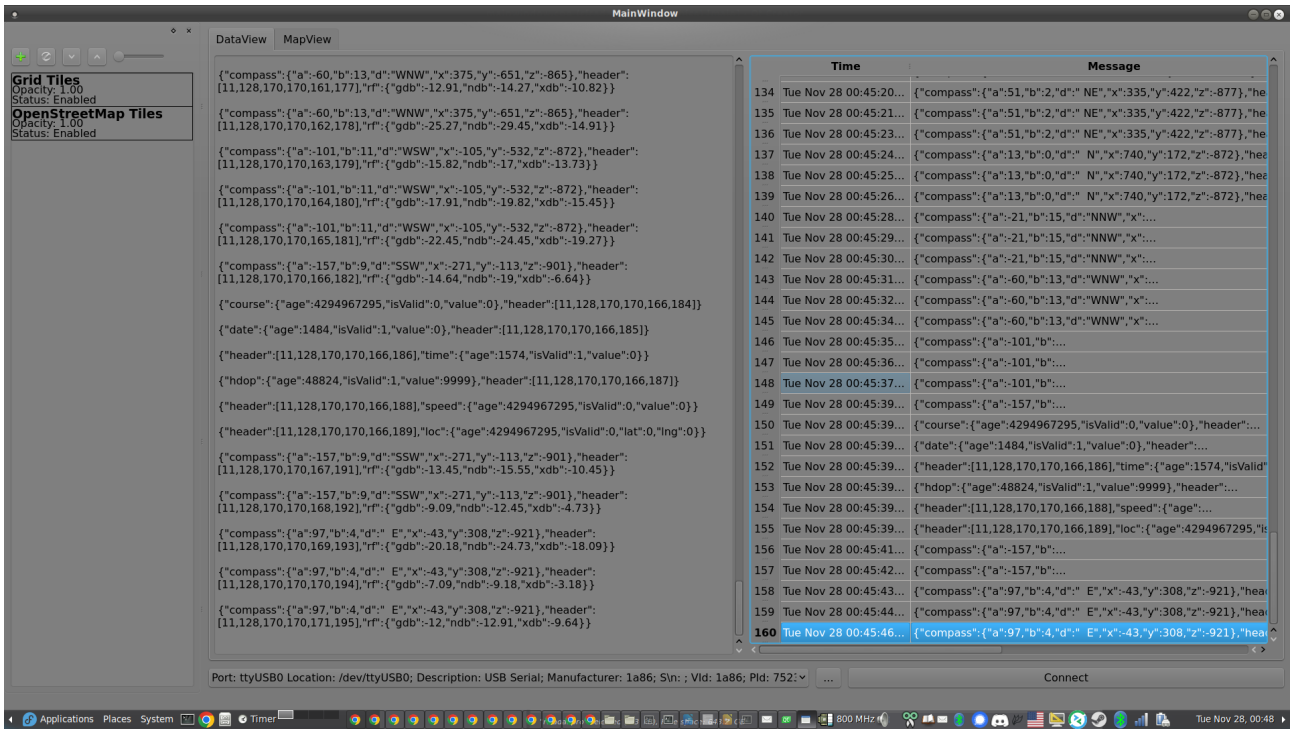



Рисунок 3.2 — Загальний вигляд програми на закладці DataView

- MapView показує мапу з індикатором на ній, загальний вигляд віджета мапи показаний на рисунку 3.3.

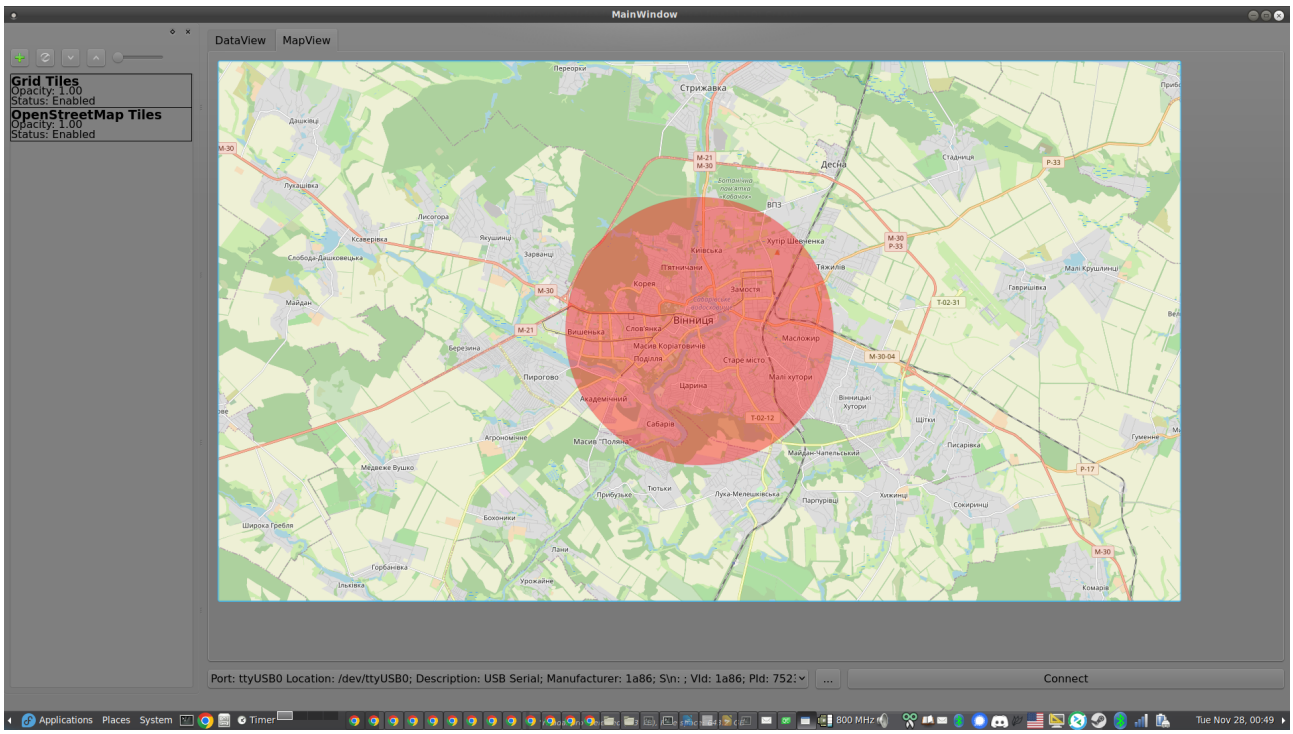



Рисунок 3.3 — Загальний вигляд програми на закладці MapView

Внизу вікна програми присутній комбобокс, що показує доступні в системі серійні пристрої. Можливе оновлення списку пристроїв кнопкою (...) оновлення яка викликає функцію updateDeviceList, що знаходиться в класі основного вікна. Поряд є кнопка Connect, після натискання якої відбувається з'єднання з пристроєм приймачем.

Взаємодія елементів програми відбувається за допомогою стандартної для QT асинхронної системи сигнал/слот.

Тобто за з'єднання відповідає слот on_connect_pushButton_clicked, у випадку від'єднання пристрою приймача спрацьовує стандартний слот втрати з'єднання.

За отримання даних від пристрою приймача відповідає сервісний клас serialportreader, де відбувається попередня обробка отриманого повідомлення, згідно протоколу описаного в розділі 3.3.

Після того як serialportreader закінчив обробку повідомлення, системою сигнал/слот ініціюється передача повідомлення класу loramessageparser, де відбувається перетворення повідомлення в правильний JSON шляхом застосування простого регулярного виразу, після чого стандартними методами QT JSON парситься в внутрішній тип даних, сигналом ініціюється вставка в лог таблицю на DataView.

В програмі використовується лише одна бібліотека стороннього розробника це MapGraphics від raptorswing, це дуже легка бібліотека з відкритими сирцями, що дозволяє запускати аплікації з вбудованими мапами навіть на деяких електронних книжках [66]. Ліцензія FreeBSD дозволяє її використання для будь-яких проектів за умови збереження посилань та ліцензійних згадувань в коді.

Інтеграція з віджетом MapGraphics відбувається в класі основного вікна.

В додатку В наведено частковий лістинг програми обробки даних системи на комп’ютері.

# 4 АНАЛІЗ РЕЗУЛЬТАТІВ

В розділі наведено огляд параметрів розробленої системи, обмежень та недоліків та можливі способи програмно апаратного покращення.

Розроблена портативна система визначення джерела радіочастотного випромінювання складається з трьох компонентів. Два програмно апаратні рішення це пристрій турель та пристрій приймач, і програмне рішення, для обробки та візуалізації виміряних даних.

Пристрій турель незважаючи на досить компактний розмір, все таки має в висоту 23cm та радіус обертання близько 17cm разом з антеною, тобто діаметр робочого простору пристрою це 34cm, вага пристрою близько 400g. Пристрій додатково потребує захисту від зовнішніх впливів, що ще додасть в розмірі, враховуючи необхідність наявності технологічного простору між антеною та корпусом, та і вазі.

Один цикл вимірювання напрямної потужності займає близько 1s, цикл сканування простору навколо, тобто один повний оберт, займає біля 25-30s, з кроком близько 15 градусів. Тобто в один оберт відбувається близько 25 напрямних вимірювань. Збільшення потужності мікроконтролера, здатне підвищити швидкість вимірювання.

Наявність одного логарифмічного перетворювача, обмежує ідентифікацію частотних діапазонів, що можна вирішити збільшенням логарифмічних перетворювачів, що правда вимагає збільшення кількості антен, а відповідно розміру і ваги продукту. Але наявність 2-3 таких пристроїв на одній турелі виглядає досить притомною ідеєю. Звісно збільшення обчислювальної потужності здатне вирішити цю проблему. Але також потрібно враховувати, що

у цьому випадку може знадобитись додаткове використання фільтрів, що теж збільшить ціну пристрою.

Наскільки б не була геніальною ідея використати підшипників у якості контактів для передачі живлення, та даних, настільки вона є абсурдною. Оскільки - так, тут є присутнє дрижання контактів під час обертання, збільшений опір, з часом може утворюватись окис. Це може бути вирішено частково використанням графітової змазки, але не виключено, що з часом все одно доведеться проводити мейнтененс системи.

В системі відсутній віддалений контроль рівня заряду батарей, це теж може бути вирішене шляхом додавання, це одного контакт підшипника, оскільки живлення відбувається через стабілізатор напруги і вимірювання напруги живлення не покаже нічого допоки не буде пізно.

Можливе встановлення даного пристрою на дрон для проведення вимірів в повітрі, але для цього однозначно необхідно збільшувати обчислювальну потужність. У цьому випадку маса пристрою може бути зменшена за рахунок використання бортового живлення, ба більше можливе використання бортового GPS після програмно апаратної модифікації.

Для системи був розроблений власний високорівневий протокол передачі даних, в якому ще присутнє місце для оптимізацій.

Пристрій приймач, розміром середнього USB flash. З'єднання з комп’ютером відбувається по серійному порту. Враховуючи, що два пристрої турель та приймач бажано щоб знаходились в прямій видимості, лише за такої умови може бути досягнута дальність 10km, це створює незручності у користуванні, або зменшення радіусу дії. Даний недолік можна усунути використання наприклад Blutooth технології, але також потрібно врахувати, що у такому випадку пристрій турель має перебувати на віддалені не менше 100 метрів від приймача, оскільки вона буде реєструвати його роботу. Цей недолік в

свою чергу можна мінімізувати програмно шлях створення фільтрів в програмі візуалізації.

Програма візуалізації на даному етапі знаходиться на початковій стадії розробки. І простору для вдосконалень тут дуже багато, від гарного радар подібного інтерфейсу, до вже згаданих програмних фільтрів, з можливістю ігнорувати певні джерела радіочастотного випромінювання.

# 5 ЕКОНОМІЧНИЙ РОЗДІЛ

## 5.1 Технологічний аудит розробленої портативної системи визначення наявності джерел радіочастотного випромінювання

Як відомо, радіочастотне випромінювання, яке є невід’ємною частиною сучасного світу, може становити потенційну загрозу для безпеки людини, оскільки може бути використане для проведення шпигунської, терористичної діяльності, тощо, а також може створювати перешкоди для санкціонованого використання радіочастотного спектру і спричиняти шкоду здоров'ю людей. Саме тому задача виявлення та локалізації джерел радіочастотного випромінювання є актуальною є задачею, розв’язання якої дозволить ідентифікувати їх джерело, характеристики та напрямок спрямування, а також вчасно прийняти заходи щодо їх нейтралізації або усунення.

Тому метою виконаної нами магістерської кваліфікаційної роботи була програмно-апаратна розробка такої системи, яка, з одного боку, була би зручною, портативною, а з іншого, могла би ефективно впоратися з викликами, які виникають, покращити методи ідентифікації пристроїв, що порушують повітряний простір та чистоту радіочастотного діапазону.

Для досягнення цієї мети нами було: вивчено наявні імплементації пристроїв визначення місцеположення об’єкта з можливістю передавання інформації з допомогою LoRa та аналізу їх недоліків; досліджено та вибрано технології та апаратні пристрої, що відповідають висунутим вимогам; розроблено технічне завдання; спроектовано апаратну та програмну частини пристрою.

В результаті були розроблені програмне і алгоритмічне забезпечення, яке може бути застосоване для забезпечення безпеки повітряного простору об’єктів, які перебувають під охороною, від впливу радіочастотного випромінювання.

Для встановлення комерційного потенціалу розробленої нами системи визначення наявності джерел радіочастотного випромінювання було запрошено 3-х експертів – кандидата технічних наук, професора Папінова В.М., кандидатів технічних наук, доцентів Маслія Р.В. та Овчинникова К.В.

Визначення комерційного потенціалу розробленої нами системи визначення наявності джерел радіочастотного випромінювання було здійснено за критеріями, наведеними в таблиці 5.1.

Таблиця 5.1 – Рекомендовані критерії оцінювання рівня комерційного потенціалу будь-якої розробки і їх бальна оцінка

| Критерії оцінювання та бали (за 5-ти бальною шкалою) | | | | | |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| Технічна здійсненність концепції: | | | | | |
| 1 | Достовірність концепції не підтверджена | Концепція підтверджена експертними висновками | Концепція підтверджена розрахунками | Концепція перевірена на практиці | Перевірено роботоздатність продукту в реальних умовах |
| Ринкові переваги (недоліки): | | | | | |
| 2 | Багато аналогів на малому ринку | Мало аналогів на малому ринку | Кілька аналогів на великому ринку | Один аналог на великому ринку | Продукт не має аналогів на великому ринку |
| 3 | Ціна продукту значно вища за ціни аналогів | Ціна продукту дещо вища за ціни аналогів | Ціна продукту приблизно дорівнює цінам аналогів | Ціна продукту дещо нижче за ціни аналогів | Ціна продукту значно нижче за ціни аналогів |
| 4 | Технічні та споживчі властивості продукту значно гірші, ніж в аналогів | Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів | Технічні та споживчі властивості продукту на рівні аналогів | Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів | Технічні та споживчі властивості продукту значно кращі, ніж в аналогів |

Продовження таблиці 5.1

| Критерії оцінювання та бали (за 5-ти бальною шкалою) | | | | | |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| Ринкові перспективи | | | | | |
| 5 | Експлуатаційні витрати значно вищі, ніж в аналогів | Експлуатаційні витрати дещо вищі, ніж в аналогів | Експлуатаційні витрати на рівні експлуатаційних витрат аналогів | Експлуатаційні витрати трохи нижчі, ніж в аналогів | Експлуатаційні витрати значно нижчі, ніж в аналогів |
| 6 | Ринок малий і не має позитивної динаміки | Ринок малий, але має позитивну динаміку | Середній ринок з позитивною динамікою | Великий стабільний ринок | Великий ринок з позитивною динамікою |
| 7 | Активна конкуренція великих компаній на ринку | Активна конкуренція | Помірна конкуренція | Незначна конкуренція | Конкурентів немає |
| Практична здійсненність | | | | | |
| 8 | Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї | Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців | Необхідне незначне навчання фахівців та збільшення їх штату | Необхідне незначне навчання фахівців | Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї |
| 9 | Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні | Потрібні незначні фінансові ресурси. Джерела фінансування відсутні | Потрібні значні фінансові ресурси. Джерела фінансування є | Потрібні незначні фінансові ресурси. Джерела фінансування є | Не потребує додаткового фінансування |

Продовження таблиці 5.1

| Критерії оцінювання та бали (за 5-ти бальною шкалою) | | | | | |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| 10 | Необхідна розробка нових матеріалів | Потрібні матеріали, що використовуються у військово-промисловому комплексі | Потрібні дорогі матеріали | Потрібні досяжні та дешеві матеріали | Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві |
| 11 | Термін реалізації ідеї більший за 10 років | Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років | Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років | Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років | Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років |
| 12 | Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту | Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу | Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу | Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту | Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту |

Запрошені нами експерти оцінили розроблену нами систему визначення наявності джерел радіочастотного випромінювання, таким чином (див. таблицю 5.2):

Таблиця 5.2 – Результати технологічного аудиту розробленого програмного продукту (за шкалою оцінювання «0»-«1»-«2»-«3»-«4»)

| Критерії | Прізвище, ініціали експертів | | |
|---|---|---|---|
| | Папінов В.М. | Маслій Р.В | Овчинников К.В. |
| | Бали, що їх виставили експерти: | | |
| 1 | 4 | 4 | 3 |
| 2 | 4 | 4 | 4 |
| 3 | 3 | 4 | 3 |
| 4 | 4 | 4 | 4 |
| 5 | 3 | 3 | 4 |
| 6 | 4 | 4 | 3 |
| 7 | 3 | 4 | 4 |
| 8 | 4 | 3 | 3 |
| 9 | 4 | 4 | 3 |
| 10 | 4 | 4 | 4 |
| 11 | 4 | 4 | 3 |
| 12 | 3 | 4 | 4 |
| Сума балів | $СБ_1 = 44$ | $СБ_2 = 46$ | $СБ_3 = 42$ |
| Середньо-арифметична сума балів $\overline{СБ}$ | $\overline{СБ} = \frac{\sum_{1}^{3} СБ_i}{3} = \frac{44+46+42}{3} = \frac{132}{3} = 44,00$ | | |

Для встановлення комерційного потенціалу розробленої нами системи визначення наявності джерел радіочастотного випромінювання скористуємося рекомендаціями, які наведено в таблиці 5.3 [67].

Таблиця 5.3 – Рівні комерційного потенціалу будь-якої наукової розробки

| Середньо-арифметична сума балів $\overline{СБ}$, розрахована на основі висновків експертів | Рівень комерційного потенціалу розробки |
|---|---|
| 0 – 10 | Низький |
| 11 – 20 | Нижче середнього |
| 21 – 30 | Середній |
| 31 – 40 | Вище середнього |
| 41 – 48 | Високий |

Оскільки середньо-арифметична сума балів, що їх виставили експерти, складає 44 бали, то це свідчить, що розроблена нами система визначення наявності джерел радіочастотного випромінювання має рівень комерційного потенціалу, який вважається «високим».

Це пояснюється тим, що в розробленій нами системі визначення наявності джерел радіочастотного випромінювання планується розроблення алгоритмів оптимізації управління модулями та режимами роботи пристрою, імплементація заходів захисту, збільшення радіусу дії та зменшення ціни готового пристрою завдяки вибору оптимальних рішень та компонентів для їх реалізації.

## 5.2 Розрахунок витрат на розроблення системи визначення наявності джерел радіочастотного випромінювання

При розробленні системи були зроблені певні витрати.

Основні з них такі:

А). Основна заробітна плата $З_о$ розробників, яка визначається за формулою:

$$З_о = \frac{М}{Т_р} t \text{ грн} \qquad (5.1)$$

де М – місячний посадовий оклад розробника, грн; приймемо, що М = (6700…25000) грн/місяць;

$Т_р$ – число робочих днів в місяці; приймемо $Т_р$ = 24 дгі;

t – число днів роботи розробників.

Зроблені розрахунки зведемо до таблиці 5.4:

Таблиця 5.4 – Основна заробітна плата розробників

| Найменування посади виконавця | Місячний посадовий оклад, грн | Оплата за робочий день, грн | Число днів роботи | Витрати на оплату праці, грн |
|---|---|---|---|---|
| 1. Науковий керівник магістерської роботи | 21820 | 909,17 | 20 годин | 3030,56≈ 3031 |
| 2. Магістрант-студент-виконавець | 2000 (беремо 6700) | 279,17 | 72 | 20100,24≈ 20100 |
| 3. Консультант з економічної частини | 19725 | 821,88 | 1,5 години | 205,46≈ 206 (при 6-годинному робочому дні) |
| Загалом | | | | $З_о$ = 23337 грн |

Б). Додаткова заробітна плата $З_д$ розробників розраховується як (10…12)% від величини їх основної заробітної плати, тобто:

$$З_д = \alpha З_о = (0,1…0,12) З_о \qquad (5.2)$$

Приймемо, що α = 0,113. Тоді для нашого випадку отримаємо:

$$З_д = 0,112 \times 23337 = 2613,74 \approx 2614 \text{ грн.}$$

В). Нарахування на заробітну плату $НЗП_{зп}$ розробників (дослідників) розраховуються за формулою:

$$НЗП_{зп} = (З_о + З_д) \cdot \frac{\beta}{100}, \quad (5.3)$$

де β – ставка обов'язкового єдиного внеску на державне соціальне страхування, %. β = 22%. Тоді:

$НЗН_{зп}$ = (23337 + 2614) × 0,22 = 5709,22 ≈ 5710 грн.

Г). Амортизація основних засобів А, які використовувались під час виконання цієї роботи:

$$A = \frac{Ц \cdot Н_а}{100} \frac{Т}{12} \text{ грн} \quad (5.4)$$

де Ц – загальна балансова вартість основних засобів, грн;

$Н_а$ – річна норма амортизаційних відрахувань. Для нашого випадку можна прийняти, що $Н_а$ = (2,5...25)%;

Т – термін використання основних засобів, місяці.

Зроблені розрахунки зведено в таблицю 5.5.

Таблиця 5.5 – Розрахунок амортизаційних відрахувань

| Найменування обладнання, приміщень тощо | Балансова вартість, грн. | Норма амортизації, % | Термін використання, міс. | Величина амортизаційних відрахувань, грн |
|---|---|---|---|---|
| 1. Комп'ютерна техніка, обладнання тощо | 79500 | 23,5 | 3,5 (при 80% використанні | 4359,25 ≈ 4360 |
| 2. Приміщення університету, кафедри | 35100 | 3,3 | 3,5 при 80% використанні | 270,27 ≈ 271 |
| Всього | | | | А = 4631 грн |

Д). Витрати на матеріали М розраховуються за формулою:

$$M = \sum_{1}^{n} H_i Ц_i K_i - \sum_{1}^{n} B_i Ц_в \text{ грн} \quad (5.5)$$

де $H_i$ – витрати матеріалу *i*-го найменування, кг; $Ц_i$ – вартість матеріалу *i*-го найменування; $K_i$ – коефіцієнт транспортних витрат, $K_i$ = (1,1…1,15); $B_i$ – маса відходів матеріалу *i*-го найменування; $Ц_в$ – ціна відходів матеріалу *i*-го найменування; n – кількість видів матеріалів.

Е). Витрати на комплектуючі К розраховуються за формулою:

$$K = \sum_{1}^{n} H_i Ц_i K_i \text{ грн} \quad (5.6)$$

де $H_i$ – кількість комплектуючих *i*-го виду, шт.; $Ц_i$ – ціна комплектуючих *i*-го виду; $K_i$ – коефіцієнт транспортних витрат, $K_i$ = (1,1…1,15); n – кількість видів комплектуючих.

Під час виконання роботи загальні витрати на матеріали та комплектуючі склали приблизно 2250 грн.

Ж). Витрати на силову електроенергію $B_e$ розраховуються за формулою:

$$B_e = \frac{B П Ф K_n}{K_д} \quad (5.7)$$

де В – вартість 1 кВт-год. електроенергії, в 2023 р. В ≈ 4,5 грн/кВт;

П – установлена потужність обладнання, кВт; П = 1,5 кВт;

Ф – фактична кількість годин роботи обладнання, годин.

Приймемо, що Ф = 345 годин;

$K_п$ – коефіцієнт використання потужності; $K_п$ < 1 = 0,64.

$K_д$ – коефіцієнт корисної дії, $K_д$ = 0,58.

Тоді витрати на силову електроенергію будуть дорівнювати:

$$В_е = \frac{В \cdot П \cdot Ф \cdot К_п}{К_д} = \frac{4,5 \cdot 1,5 \cdot 345 \cdot 0,64}{0,58} = 2569,66 \approx 2570 \text{ грн.}$$

И). Інші витрати $В_{інш}$ можна прийняти як (50…300)% від основної заробітної плати розробників, тобто:

$$В_{інш} = (0,5....3) \times З_о. \quad (5.8)$$

Для нашого випадку отримаємо:

$В_{інш} = 1,09 \times 23337 = 25437,33 \approx 25438$ грн.

К). Сума всіх попередніх статей витрат складає витрати на виконання роботи безпосередньо розробником-магістрантом – В.

В = 23337+ 2614 + 5710+ 4631 + 2250 + 2570 + 25438 = 66550 грн.

Л). Загальні витрати на розроблення програмного забезпечення $В_{заг}$ розраховуються за формулою:

$$В_{заг} = \frac{В}{\beta} \quad (5.9)$$

де $\beta$ – коефіцієнт, який характеризує етап (стадію) виконання цієї роботи. Можна прийняти, що, $\beta \approx 0,95$ [67], оскільки робота практично завершена.

Тоді: $В_{заг} = \frac{66550}{0,96} = 70052,63$ грн або приблизно 70 тисяч грн.

Тобто прогнозовані загальні витрати на розробку системи визначення наявності джерел радіочастотного випромінювання становлять приблизно 70 тисячі грн.

## 5.3 Розрахунок економічного ефекту від можливої комерціалізації розробленої системи визначення наявності джерел радіочастотного випромінювання

Економічний ефект від впровадження та можливої комерціалізації розробленої нами системи визначення наявності джерел радіочастотного випромінювання пояснюється використанням алгоритмів оптимізації управління модулями та режимами роботи пристрою, імплементацією заходів захисту, збільшенням радіусу дії та відносним зменшенням ціни готового пристрою завдяки вибору оптимальних рішень та компонентів для його реалізації.

Тому нашу розробку можна реалізовувати на ринку дещо дешевше, ніж аналогічні за функціями розробки (що суттєво збільшить попит на неї). Так, якщо подібні системи визначення наявності джерел радіочастотного випромінювання у 2022-2023 роках коштували на ринку приблизно 4 тисячі грн, то нашу розробку можна буде реалізовувати на ринку дещо дешевше, приблизно за 3,9 тисяч грн.

Аналіз місткості ринку показує, що на сьогодні в Україні попит на подібну розробку, особливо в умовах воєнного стану, може бути великим. Тому можна очікувати стрімке (!) зростання попиту на нашу розробку принаймні протягом 3-х років після її впровадження (до появи нових, більш ефективних розробок) .

Тобто, якщо наша  розробка буде впроваджена з 1 січня 2024 року, то її результати будуть виявлятися протягом 2024-го, 2025-го та 2026-го років.

Прогноз зростання попиту на нашу розробку складає по роках:

2023 рік – базовий попит 100 шт.

а) 2024 р. – приблизно +100 шт. до базового року;

б) 2025 р. – +200 шт. до базового року;

в) 2026 р. – +300 шт. до базового року.

Можливе збільшення чистого прибутку $\Delta П_і$, що його може отримати потенційний інвестор від комерціалізації, тобто виведення нашої розробки на ринок, становитиме:

$$\Delta П_і = \sum_{1}^{n} (\Delta Ц_о N + Ц_о \Delta N)_i \lambda \rho (1 - \frac{\upsilon}{100}) \quad (5.10)$$

де $\Delta Ц_о$ – покращення основного якісного показника від впровадження результатів нашої розробки у цьому році. Для нашого випадку це є зменшення ціни реалізації нашої розробки $\Delta Ц_о$ = (3,9 – 4,0) = - 0,1 тисяч грн;

N – основний кількісний показник, який визначає обсяг діяльності у році до впровадження результатів розробки; N = 100 шт.;

$\Delta N$ – покращення (збільшення основного кількісного показника від впровадження результатів розробки.

Таке збільшення становитиме: у 2024 році – + 100 шт., у 2025 році – +200 шт., у 2026 році – + 300 шт. (відносно базового 2023 року);

$Ц_о$ – основний якісний показник (тобто ціна), який визначає обсяг діяльності у році після впровадження результатів розробки; $Ц_о$= 3,9 тисячі грн;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки; для нашого випадку n = 3;

$\lambda$ – коефіцієнт, який враховує сплату податку на додану вартість; $\lambda = 0{,}8333$;

$\rho$ – коефіцієнт, який враховує рентабельність продукту. Рекомендується приймати $\rho$= (0,2...0,5); візьмемо $\rho$= 0,5;

$\upsilon$ – ставка податку на прибуток. У 2023 і наступних роках $\upsilon$ = 18%.

Тоді можливе зростання чистого прибутку $\Delta П_1$ для потенційного інвестора протягом першого року від можливого впровадження нашої розробки (2024 р.) складе:

$$\Delta П_1 = [-0,1 \cdot 100 + 3,9 \cdot 100] \cdot 0,8333 \cdot 0,5 \cdot (1 - \frac{18}{100}) \approx 130 \text{ тис. грн.}$$

Можливе зростання чистого прибутку $\Delta П_2$ для потенційного інвестора від можливого впровадження нашої розробки протягом другого (2025 р.) року складе:

$$\Delta П_2 = [-0,1 \cdot 100 + 3,9 \cdot 200] \cdot 0,8333 \cdot 0,5 \cdot (1 - \frac{18}{100}) \approx 263 \text{ тис. грн.}$$

Можливе зростання чистого прибутку $\Delta П_3$ для потенційного інвестора від можливого впровадження нашої розробки протягом третього (2026 р.) року складе:

$$\Delta П_3 = [-0,1 \cdot 100 + 3,9 \cdot 300] \cdot 0,8333 \cdot 0,5 \cdot (1 - \frac{18}{100}) \approx 397 \text{ тис. грн.}$$

Приведена вартість зростання всіх чистих прибутків від можливого впровадження нашої розробки становитиме:

$$ПП = \sum_{1}^{m} \frac{\Delta П_i}{(1+\tau)^t} \qquad (5.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої роботи, грн;

т – період часу, протягом якого виявляються результати впровадженої роботи, роки. Для нашого випадку т = 3 роки;

$\tau$ – ставка дисконтування. Приймемо $\tau$ = 0,10 (10%);

t – період часу від моменту початку розроблення нашої системи визначення наявності джерел радіочастотного випромінювання до моменту отримання можливих чистих прибутків потенційним інвестором.

Тоді приведена вартість зростання всіх можливих чистих прибутків ПП, що їх може отримати потенційний інвестор від комерціалізації нашої розробки, складе:

$$\text{ПП} = \frac{130}{(1+0,1)^2} + \frac{263}{(1+0,1)^3} + \frac{397}{(1+0,1)^4} \approx 107 + 198 + 271 = 576 \text{ тисяч грн.}$$

Теперішня вартість інвестицій PV, що повинні бути вкладені для реалізації нашої розробки: $PV = (1,0…5,0) \times B_{заг}$.

Для нашого випадку $PV = (1,0…5,0) \times 70 = 1,5 \times 70 = 105$ тисяч грн.

Абсолютний ефект від можливих вкладених інвестицій $E_{абс}$.

$$E_{абс} = \text{ПП} - PV, \qquad (5.12)$$

де ПП – приведена вартість збільшення всіх чистих прибутків для інвестора від можливого впровадження нашої розробки, грн;

PV – теперішня вартість інвестицій PV = 376 тисяч грн.

Абсолютний ефект від можливого впровадження нашої розробки (при прогнозованому ринку збуту) за три роки складе:

$$E_{абс} = 576 - 105 = 471 \text{ тисяч грн.}$$

Оскільки $E_{абс} > 0$, то комерціалізація нашої розробки може бути доцільною.

Далі розрахуємо внутрішню дохідність $E_в$ вкладених інвестицій:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1 \qquad (5.13)$$

де $E_{абс}$ – абсолютний ефект вкладених інвестицій; $E_{абс}$ = 471 тис. грн;

PV –теперішня вартість початкових інвестицій PV = 105 тис. грн;

$Т_ж$ – життєвий цикл розробки, роки.

$Т_ж$ = 4 років (2023-й, 2024-й, 2025-й, 2026-й роки)

Для нашого випадку отримаємо:

$$Е_в = \sqrt[4]{1 + \frac{471}{105}} - 1 = \sqrt[4]{1 + 4,4857} - 1 = \sqrt[4]{5,4857} - 1 = 1,53 - 1 = 0,53 = 53\%.$$

Далі визначимо ту мінімальну дохідність, нижче за яку потенційному інвестору не вигідно буде займатися комерціалізацією нашої розробки.

Мінімальна дохідність або мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою:

$$\tau_{мін} = d + f, \qquad (5.14)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d = (0,10...0,12);

f – показник, що характеризує ризикованість вкладень; f = (0,05...0,30).

Для нашого випадку отримаємо:

$$\tau_{мін} = 0,12 + 0,30 = 0,42 \text{ або } \tau_{мін} = 42\%.$$

Оскільки величина $Е_в$ = 54% > $\tau_{мін}$ = 42%, то потенційний інвестор у принципі може бути зацікавлений у фінансуванні та комерціалізації розробленої нами системи визначення наявності джерел радіочастотного випромінювання.

Далі розраховуємо термін окупності коштів, вкладених у можливу комерціалізацію розробленої нами системи визначення наявності джерел радіочастотного випромінювання.

Термін окупності $Т_{ок}$ розраховується за формулою:

$$T_{ок} = \frac{1}{E_в}. \qquad (5.15)$$

Для нашого випадку термін окупності $T_{ок}$ коштів становитиме:

$$T_{ок} = \frac{1}{0,53} = 1,89 \text{ років} < 3 \text{ років},$$

що свідчить про потенційну доцільність комерціалізації розробленої нами системи визначення наявності джерел радіочастотного випромінювання.

Далі проведено моделювання залежності величини внутрішньої дохідності вкладених потенційним інвестором коштів в комерціалізацію розробленої нами системи визначення наявності джерел радіочастотного випромінювання від рівня інфляції в країні.

Так, якщо рівень інфляції в країні зросте до 20%, то:

$$ПП = \frac{130}{(1+0,2)^2} + \frac{263}{(1+0,2)^3} + \frac{397}{(1+0,2)^4} \approx 90 + 152 + 191 = 433 \text{ тисяч грн.}$$

Тоді абсолютний ефект від можливого впровадження нашої розробки за три роки складе:

$$Е_{абс} = 433 - 105 = 328 \text{ тисяч грн.}$$

Внутрішня дохідність $Е_в$ диввись формулу 5.16.

$$E_в = \sqrt[T_{ж}]{1 + \frac{E_{абс}}{PV}} - 1 \qquad (5.16)$$

де $Е_{абс}$ – абсолютний ефект вкладених інвестицій; $Е_{абс}$ = 328 тисяч грн;

PV –теперішня вартість початкових інвестицій PV = 105 тисяч грн.

Для нашого випадку отримаємо:

$$Е_в = \sqrt[4]{1 + \frac{328}{105}} - 1 = \sqrt[4]{1 + 3,1238} - 1 = \sqrt[4]{4,1238} - 1 = 1,425 - 1 = 0,425 = 42,5\%.$$

Оскільки величина $Е_в$ = 42,5% $> \tau_{мін}$ = 42%, то потенційний інвестор у принципі також може бути зацікавлений у фінансуванні та комерціалізації нашої розробки.

Прийнявши рівень інфляції у 30% отримаємо:

$$ПП = \frac{130}{(1+0,3)^2} + \frac{263}{(1+0,3)^3} + \frac{397}{(1+0,3)^4} \approx 70 + 120 + 139 = 329 \text{ тисяч грн.}$$

Тоді абсолютний ефект від можливого впровадження нашої розробки складе:

$Е_{абс}$ = 329 – 105 = 224 тисяч грн.

Внутрішня дохідність $Е_в$ дивись формулу 5.17.

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1 \qquad (5.17)$$

де $Е_{абс}$ – абсолютний ефект вкладених інвестицій; $Е_{абс}$ = 224 тисяч грн;

PV –теперішня вартість початкових інвестицій PV = 105 тисяч грн.

Для нашого випадку отримаємо:

$$Е_в = \sqrt[4]{1 + \frac{224}{105}} - 1 = \sqrt[4]{1 + 2,1333} - 1 = \sqrt[4]{3,1333} - 1 = 1,33 - 1 = 0,33 = 33\%.$$

Оскільки величина $Е_в$ = 33% $< \tau_{мін}$ = 42%, то потенційний інвестор у принципі може бути і незацікавлений у фінансуванні та комерціалізації нашої розробки, але остаточне рішення щодо цього питання буде прийматися при врахуванні інших обставин (наприклад, шляхом зниження рівня прийнятого ризику з f = 30% до меншої величини, або шляхом підняття ціни реалізації нашої розробки тощо).

Зроблені розрахунки у вигляді графіків наведено на рис. 5.1.

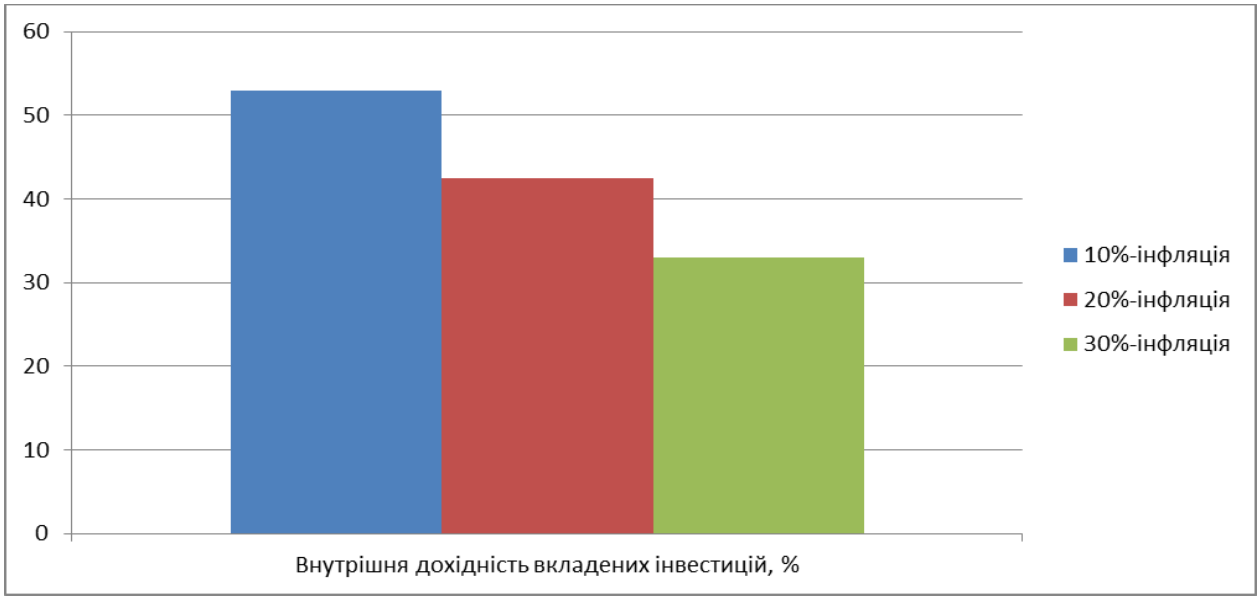



Рисунок 5.1 – Моделювання залежності величини внутрішньої дохідності потенційних інвестицій від рівня інфляції в країні при її значеннях у 10%, 20% і 30%

Аналіз графіка на рис 5.1 показує, що при рівні інфляції в 10% величина внутрішньої дохідності інвестицій становить $Е_в$ = 53%, що значно більше порогового значення $\tau_{мін}$ = 42% і тому комерціалізація нашої розробки може бути доцільною. При рівні інфляції в 20% величина внутрішньої дохідності інвестицій, вкладених в комерціалізацію нашої розробки, становить $Е_в$ = 42,5%, що також більше порогового значення $\tau_{мін}$ = 42%, і тому комерціалізація нашої розробки може бути для інвестора доцільною. При інфляції більше, ніж у 30% величина внутрішньої дохідності інвестицій $Е_в$ = 33%, вкладених в комерціалізацію нашої розробки, становить менше порогового значення $\tau_{мін}$ = 42%. Тому за таких умов комерціалізація розробленої нами системи визначення наявності джерел радіочастотного випромінювання від рівня може бути проблематичною і потребує проведення додаткових розрахунків.

Результати виконаної економічної частини магістерської кваліфікаційної роботи наведено в таблиці 5.6.

Таблиця 5.6 – Результати економічної частини

| Показники | Задані у ТЗ | Досягнуті у магістерській кваліфікаційній роботі | Висновок |
|---|---|---|---|
| 1. Витрати на розробку | Не більше 100 тис. грн | 70 тис. грн. | Досягнуто |
| 2. Абсолютний ефект від впровадження розробки, тисяч грн | Не менше 400 тисяч грн | 431 тисяч грн (при 10%-інфляції) | Виконано |
| 3. Внутрішня дохідність інвестицій, % | не менше 42% | 53% (при 10%-інфляції) | Досягнуто |
| 4. Термін окупності інвестицій, роки | до 3-ти років | 1,89 років | Виконано |

Таким чином, основні техніко-економічні показники розробленої нами системи визначення наявності джерел радіочастотного випромінювання, визначені у технічному завданні, виконані.

# ВИСНОВКИ

В роботі було проведено дослідження предметної області принципів детекції радіочастотних джерел, методів та рішень детекції, що існують на ринку. Дослідження технологій розробки програмно-апаратного забезпечення для цих цілей, аналіз діапазонів робочих частот БПЛА, та технологій передачі даних на великі відстані, а також вибір програмних засобів для розробки компонентів системи. Запропоновано власний підхід для реалізації системи визначення джерел радіочастотного випромінювання.

На основі запропонованого підходу реалізовано систему, з використанням обраних програмно апаратних рішень, для створення використовувались такі програмні продукти як kicad, FreeCad, QtCreator, Arduino IDE, а також апаратні засоби як TinySA, NanoVNA. Основною мовою реалізації програмних рішень використано C++. Розроблена система відповідає мінімальним вимогам поставленим в задачі здатна визначати наявність джерел радіочастотного випромінювання в діапазоні від 2 GHz до 5,8 GHz, сканування одного оберту відбувається менше ніж за 1 хвилину, віддаленість елементів системи одна від одної не більше 10 км. Як вже було згадано один цикл вимірювання напрямної потужності займає близько 1s, цикл сканування простору навколо, тобто один повний оберт, займає біля 25-30s, з кроком близько 15 градусів. Тобто в один оберт відбувається близько 25 напрямних вимірювань. Ціна пристрою біля 3000 грн. Збільшення потужності мікроконтролера, здатне підвищити швидкість вимірювання звісно, це буде викликати збільшення ціни пристрою. Система має потенціал для подальшого розвитку і покращення

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Електромагнітне випромінювання  Види електромагнітних коливань URL: https://uk.wikipedia.org/wiki/Електромагнітне_випромінювання (дата звернення 5.10.2023)
2. Unmanned aerial vehicle URL: https://en.wikipedia.org/wiki/Unmanned_aerial_vehicle (дата звернення 5.10.2023)
3. Радіолокація URL: https://uk.wikipedia.org/wiki/Радіолокація (дата звернення 5.10.2023)
4. Пасивна радіолокація URL: https://uk.wikipedia.org/wiki/Пасивна_радіолокація (дата звернення 5.10.2023)
5. How GPS trackers can save your life URL: https://www.gpswox.com/en/blog/useful-information/how-gps-tracker-can-save-your-life-in-2020 (дата звернення 5.10.2023)
6. Global Positioning System URL: https://en.wikipedia.org/wiki/Global_Positioning_System (дата звернення 5.10.2023)
7. LoRa Alliance URL:  https://lora-alliance.org/ (дата звернення 5.10.2023)
8. LoRa  URL: https://en.wikipedia.org/wiki/LoRa (дата звернення 5.10.2023)
9. Phat Thai, Sameer Alam, Nimrod Lilith, Binh Nguyen. Small Flying Object Detection and Tracking in Digital Airport Tower through Spatial-Temporal ConvNets. Nanyang Technological University, Ho Chi Minh City University of Science, 2023 URL:

https://assets.researchsquare.com/files/rs-2562253/v1/e0e2f963666802c9e91541f5.pdf?c=1677677781 (дата звернення 5.11.2023)

10. Yimiao Sun, Weiguo Wang, Luca Mottola, Ruijin Wang, Yuan He1. AIM: Acoustic Inertial Measurement for Indoor Drone Localization and Tracking. Tsinghua University, 2Politecnico di Milano, Italy and RI.SE Sweden 3University of Electronic Science and Technology of China. 2022 URL: https://dl.acm.org/doi/pdf/10.1145/3560905.3568499 (дата звернення 5.11.2023)
11. Jawad Yousaf, Huma Zia, Marah Alhalabi, Maha Yaghi, Tasnim Basmaji, Eiman Al Shehhi, Abdalla Gad, Mohammad Alkhedher, Mohammed Ghazal. Drone and Controller Detection and Localization: Trends and Challenges. Abu Dhabi University, Abu Dhabi 59911, United Arab Emirates. 2022. URL: https://www.mdpi.com/2076-3417/12/24/12612 (дата звернення 5.11.2023)
12. HackRF One URL: https://hackrf.readthedocs.io/en/latest/hackrf_one.html (дата звернення 5.10.2023)
13. Tiny SA URL: https://www.tinysa.org/wiki/ (дата звернення 5.10.2023)
14. tiny SA Ultra price URL: https://vi.aliexpress.com/item/1005005009499031.html (дата звернення 5.10.2023)
15. Phuc Nguyen, Taeho Kim, Jinpeng Miao, Daniel Hesselius, Erin Kenneally, Daniel Massey, Eric Frew, Richard Han and Tam V. Towards RF-based Localization of a Drone and Its Controller, University of Colorado Boulder, *United States Department of Homeland Security, 2019 URL: **https://home.cs.colorado.edu/~rhan/Papers/2019_DRONET_RF-based_Localization.pdf** (дата звернення 5.11.2023)
16. Негода Нікіта Володимирович «Портативна радіолокаційна система виявлення та пеленгації джерел електромагнітного випромінювання

БПЛА» Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» URL: https://ela.kpi.ua/bitstream/123456789/59857/1/NehodaNV_bakalavr.pdf (дата звернення 5.10.2023)

17. Phuc Nguyen, Hoang Truong, Mahesh Ravindranathan, Anh Nguyen, Richard Han, Tam Vu. Matthan: Drone Presence Detection by Identifying Physical Signatures in the Drone's RF Communication. University of Colorado, Denver, University of Colorado, Boulder, 2017 URL: http://mnslab.org/tamvu/paper/2017_Mobisys_Drone.pdf (дата звернення 5.11.2023)

18. Yuhong Wang * , Yonghong Zeng , Sumei Sun, Peng Hui Tan, Yugang Ma and Ernest Kurniawan. Drone Controller Localization Based on RSSI Ratio. Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR). 2023 URL: **https://www.semanticscholar.org/reader/7e37cab57be7842c28a4ff56b8cdcfdf71d3a485** (дата звернення 5.11.2023)

19. РБ-301Б «Борисоглєбськ-2» URL: https://uk.wikipedia.org/wiki/РБ-301Б_«Борисоглєбськ-2» (дата звернення 5.10.2023)

20. logarithmic amplifiers URL: https://www.analog.com/en/analog-dialogue/articles/logarithmic-amplifiers-explained.html (дата звернення 5.10.2023)

21. RP2040 VS ATMEGA328 VS ESP32 VS STM32 URL: https://www.utmel.com/components/rp2040-vs-atmega328-vs-esp32-vs-stm32-comparison-some-parameters-of-microcontrollers?id=1471 (дата звернення 5.10.2023)

22. List of Arduino boards and compatible systems URL: https://en.wikipedia.org/wiki/List_of_Arduino_boards_and_compatible_systems (дата звернення 5.10.2023)

23. ESP32 URL: https://en.wikipedia.org/wiki/ESP32 (дата звернення 5.10.2023)

24. STM32 Nucleo URL: https://www.st.com/en/ecosystems/stm32-nucleo.html (дата звернення 5.10.2023)

25. STM32 development boards URL: https://stm32-base.org/boards/ (дата звернення 5.10.2023)

26. Integrated Development Environment for STM32 URL: https://www.st.com/en/development-tools/stm32cubeide.html (дата звернення 5.10.2023)

27. stm32duino URL: https://github.com/stm32duino/Arduino_Core_STM32 (дата звернення 5.10.2023)

28. Best RP2040 Boards 2023 URL: https://www.tomshardware.com/best-picks/best-rp2040-boards (дата звернення 5.10.2023)

29. Install Thonny URL: https://projects.raspberrypi.org/en/projects/getting-started-with-the-pico/2 (дата звернення 5.10.2023)

30. openlrs — About.wiki URL: https://code.google.com/archive/p/openlrs/wikis/About.wiki (дата звернення 5.10.2023)

31. openlrs - About.wiki URL: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/the-book/packet-size-considerations/#bit_packing (дата звернення 5.10.2023)

32. Udita Bhattacherjee, Ender Ozturk, Ozgur Ozdemir, Ismail Guvenc, Mihail L. Sichitiu, Huaiyu Dai. Department of Electrical and Computer Engineering,

North Carolina State University, Raleigh, NC. 2021 URL: https://www.semanticscholar.org/reader/2391cecf0c8778aef0358a911664d4f522f0900b (дата звернення 5.11.2023)

33. DJI Mavic 3 Enterprise URL: https://enterprise.dji.com/mavic-3-enterprise/specs (дата звернення 5.10.2023)
34. EVO Max 4T URL: https://auteldronesbaltic.com/en/enterprise-drones/evo-max-4t/ (дата звернення 5.10.2023)
35. ExpressLRS vs TBS Crossfire: The Best Radio Control Link for FPV Drones URL: https://oscarliang.com/expresslrs/ (дата звернення 5.10.2023)
36. FPV frequencies Management Tips and Best Practices URL:https://dronenodes.com/fpv-frequencies-management/ (дата звернення 5.10.2023)
37. FPV Frequency Reference Chart URL: https://www.getfpv.com/learn/fpv-essentials/fpv-frequency-reference-chart/ (дата звернення 5.10.2023)
38. Орлан-10 URL: https://uk.wikipedia.org/wiki/Орлан-10(дата звернення 5.10.2023)
39. Цукорок Детектори ворожих дронів FAQ URL: https://www.facebook.com/groups/uavopengroup/posts/6491050984307136/ (дата звернення 5.10.2023)
40. Про затвердження Національної таблиці розподілу смуг радіочастот України URL: https://zakon.rada.gov.ua/laws/show/1208-2005-%D0%BF#Text (дата звернення 5.10.2023)
41. РЕГЛАМЕНТ аматорського радіозв'язку України URL: https://zakon.rada.gov.ua/laws/show/z1106-23#Text (дата звернення 5.10.2023)

42. РЕГЛАМЕНТ аматорського радіозв’язку URL: https://ips.ligazakon.net/document/RE40162?an=8 (дата звернення 5.10.2023)

43. Semtech SX1278 URL: https://www.semtech.com/products/wireless-rf/lora-connect/sx1278 (дата звернення 5.10.2023)

44. SX1278 datasheet URL: https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/2R0000001Rc1/QnUuV9TviODKUgt_rpBlPz.EZA_PNK7Rpi8HA5..Sbo (дата звернення 5.10.2023)

45. macOS URL: https://en.wikipedia.org/wiki/MacOS (дата звернення 5.10.2023)

46. iOS URL: https://en.wikipedia.org/wiki/IOS (дата звернення 5.10.2023)

47. Android URL: https://en.wikipedia.org/wiki/Android_(operating_system) (дата звернення 5.10.2023)

48. Microsoft Windows URL: https://en.wikipedia.org/wiki/Microsoft_Windows (дата звернення 5.10.2023)

49. Malicious npm packages caught installing remote access trojans URL: https://www.zdnet.com/article/malicious-npm-packages-caught-installing-remote-access-trojans/ (дата звернення 5.10.2023)

50. Java_(programming_language) URL: https://en.wikipedia.org/wiki/Java_(programming_language) (дата звернення 5.10.2023)

51. Java_(software_platform) URL: https://en.wikipedia.org/wiki/Java_(software_platform) (дата звернення 5.10.2023)

52. Qt URL: https://en.wikipedia.org/wiki/Qt_(software) (дата звернення 5.10.2023)

53. solidworks oficial site URL: https://www.solidworks.com/ (дата звернення 5.10.2023)

54. solidworks URL:https://en.wikipedia.org/wiki/SolidWorks (дата звернення 5.10.2023)

55. OpenSCAD URL: https://openscad.org/ (дата звернення 5.10.2023)

56. FreeCAD homepage URL: https://www.freecad.org/ (дата звернення 5.10.2023)

57. FreeCAD URL: https://en.wikipedia.org/wiki/FreeCAD (дата звернення 5.10.2023)

58. DRV8833 Datasheet URL: https://www.ti.com/lit/ds/symlink/drv8833.pdf?ts=1700015134892 (дата звернення 21.11.2023)

59. AD8317 Datasheet URL: https://www.analog.com/media/en/technical-documentation/data-sheets/ad8317.pdf (дата звернення 21.11.2023)

60. LM4040 Datasheet URL: https://www.ti.com/lit/ds/symlink/lm4040.pdf?ts=1700493862425&ref_url=https%253A%252F%252Fwww.google.com%252F (дата звернення 24.11.2023)

61. LoRa Ra-01 specification URL: https://docs.ai-thinker.com/_media/lora/docs/ra-01s_specification.pdf (дата звернення 24.11.2023)

62. XT2108 (HM1549) datasheet URL: https://www.hmsemi.com/downfile/HM1549.PDF (дата звернення 25.11.2023)

63. Georgia Lykou, Dimitrios Moustakas, Dimitris Gritzalis «Defending Airports from UAS: A Survey on Cyber-Attacks and Counter-Drone Sensing

Technologies» URL: https://www.mdpi.com/1424-8220/20/12/3537 (дата звернення 25.11.2023)

64. PAUL ISAAC DEFFENBAUGH, 3D PRINTED ELECTROMAGNETIC TRANSMISSION AND ELECTRONIC STRUCTURES FABRICATED ON A SINGLE PLATFORM USING ADVANCED PROCESS INTEGRATION TECHNIQUES, M.S.E.E., Department of Electrical and Computer Engineering, 2014 URL: https://web.archive.org/web/20170312174308/http://emlab.utep.edu/pdfs/Deffenbaugh_PhD_Dissertation.pdf (дата звернення 25.11.2023)
65. Arduino Cryptography Library Performance URL: https://rweather.github.io/arduinolibs/crypto.html (дата звернення 27.11.2023)
66. MapGraphics URL: https://github.com/raptorswing/MapGraphics (дата звернення 28.11.2023)
67. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт. / Укладачі В.О. Козловський, О.Й. Лесько, В.В.Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

# ДОДАТКИ

# Додаток А

## (обов'язковий)

## Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ЗАТВЕРДЖУЮ
Завідувач кафедри АІІТ
____ д.т.н., проф. Олег БІСІКАЛО
«__» _____2023 року

ТЕХНІЧНЕ ЗАВДАННЯ
на магістерську кваліфікаційну роботу

ПОРТАТИВНА СИСТЕМА ВИЗНАЧЕННЯ НАЯВНОСТІ ДЖЕРЕЛА РАДІОЧАСТОТНОГО ВИПРОМІНЮВАННЯ

08-31.МКР.010.02.000 ТЗ

Керівник: к.т.н., доц. каф. АІІТ
_____ Володимир ГАРМАШ
«__» _____ 2023 р.
Розробив студент гр. 1АКІТ-22м
_____Юрій КУЗІН
«__» _____ 2023 р.

Вінниця 2023

1. Назва та галузь застосування.
Портативна система визначення наявності джерела радіочастотного випромінювання. Автоматизація та приладобудування.
2. Підстава для проведення робіт.
Підставою для виконання роботи є наказ №_247_ по ВНТУ від «_18_» ___09___ 2023р., та індивідуальне завдання на МКР, затверджене протоколом №__ засідання кафедри АІІТ від «__» _____ 2023р.
3. Мета та призначення роботи.
Метою роботи є покращити наявні методи ідентифікації пристроїв радіовипромінювання, що порушують повітряний простір та чистоту радіочастотного діапазону.
4. Джерела розробки:
- ISO 5457:1999 Technical product documentation Sizes and layout of drawing sheets;
- ISO 7200:2004 Technical product documentation Data fields in title blocks and document headers.
5. Показники призначення
Основні технічні вимоги та мінімальні системні вимоги до програми:
5.1 Структура системи
- пристрій вимірювання потужності радіочастотного сигналу та передавання результатів на відстань;
- пристрій приймач виміряних даних;
- програма для відображення даних на моніторі комп’ютера.
5.2 Вимоги до апаратної частини
- частотний діапазон чутливості 2,4-5,8 GHz;
- максимальна відстань компонентів системи 10 км;
- період обертання менше 1 хв.
5.3 Вимоги до програмної платформи
5.3.1 Вимоги до програмної платформи
- Windows, Linux, MacOS;
- Arduino IDE;
- QT framework.
5.3.2 Умови експлуатації системи
- пристрій для вимірювання повинен використовуватись за відсутності забудови, бажано на підвищенні;
- пристрій для вимірювання та пристрій приймач бажано щоб знаходились в прямій видимості;
- пристрій приймач потребує наявності USB порту.

Вихідні дані для проведення робіт:
стек технологій для розробки продукту; технічна документація та специфікація елементної бази; регламенти частот та частотні характеристики джерел радіовипромінювання

Методи дослідження:
В даній роботі використовуються лабораторні вимірювання та експериментальні методи.

Результати роботи програми _____
_____

6. Економічні показники
До економічних показників входять:
- витрати на розробку _____3000 грн_____
- приведена вартість прибутку за _2_ роки _____263 тис. грн_____-
- мінімальна дохідність _____42%_____
- термін окупності_____менше трьох років_____

7. Стадії розробки:
а) Аналіз предметної області ____ – ____
б) Вибір методів імплементації та елементної бази ____ – ____
в) Розробка схеми електричної принципової ____ – ____
г) Розробка програмного забезпечення ____ – ____
д) Економічна частина ____ – ____
е) Оформлення матеріалів до захисту МКР ____ – ____

8. Порядок контролю та приймання

Рубіжний контроль провести до «___» _____ 2023 р.
Попередній захист МКР провести «___» _____ 2023 р.
Захист МКР провести до «___» _____ 2023 р.

Розробив студент групи 1АКІТ-22м _____ Юрій КУЗІН

# Додаток Б

**(обов’язковий)**

**Ілюстративна частина**

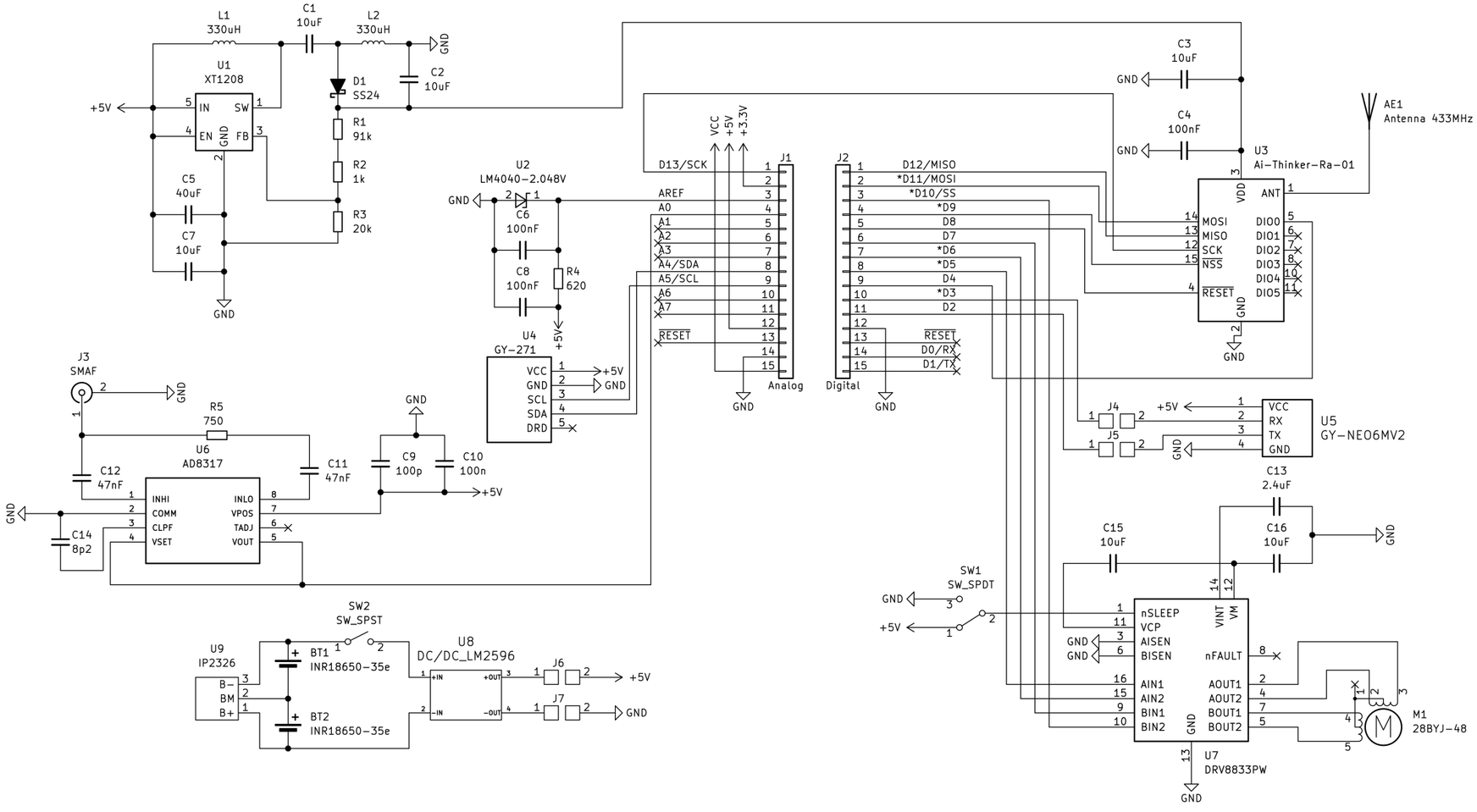



Рисунок Б.1 — Схема електрична принципова пристрою турелі

Продовження додатку Б

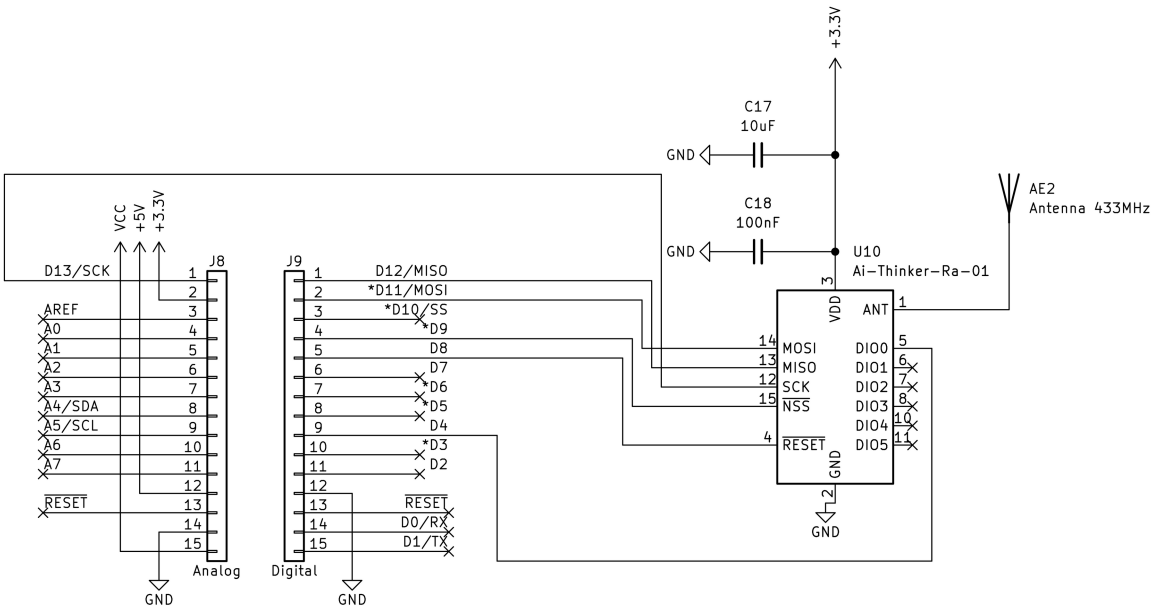
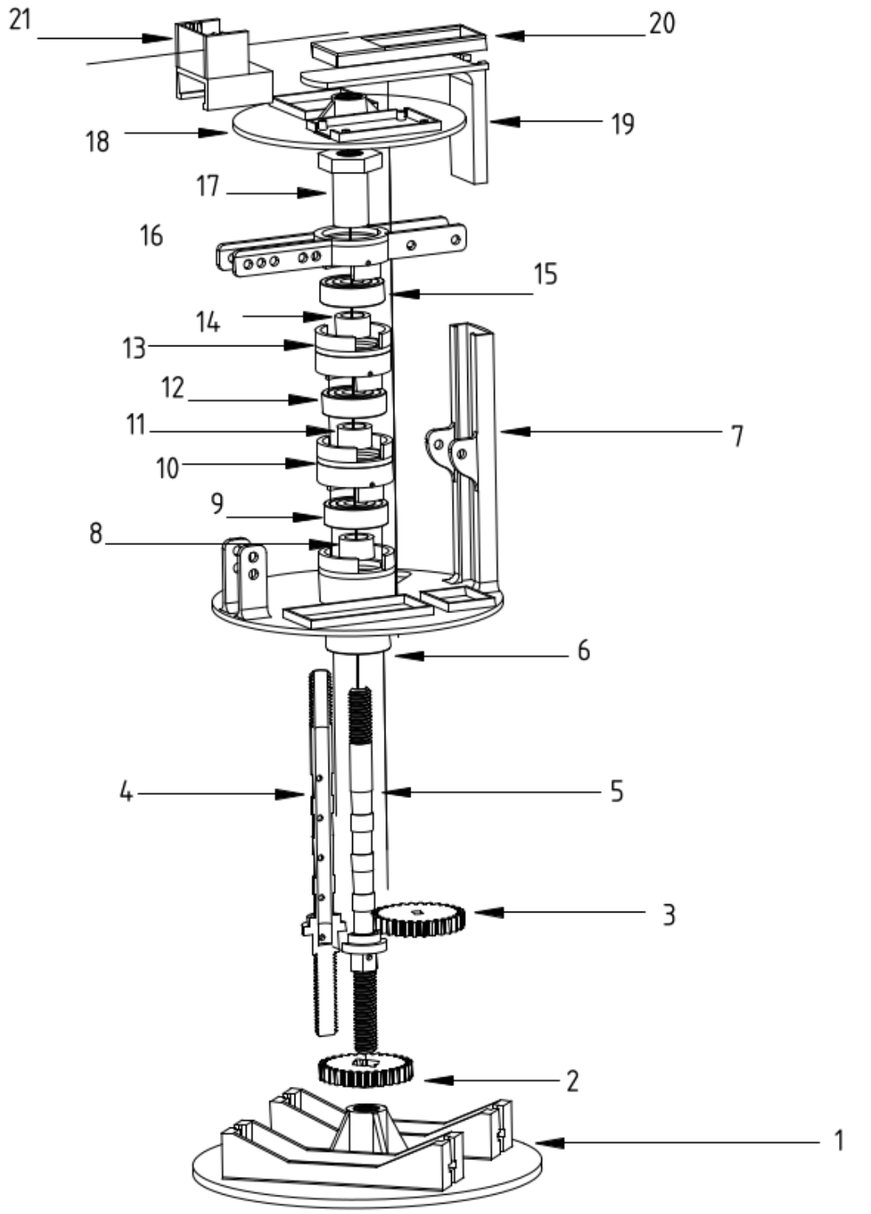



Рисунок Б.2 — Схема електрична принципова пристрою приймача




| | |
|---|---|
| 1 | Bottom platform |
| 2 | Rod gear |
| 3 | Mottor gear |
| 4 | Inner rod part with holes |
| 5 | Inner rod no holes side |
| 6 | Bearing 608zz |
| 7 | Rotatable platform |
| 8 | Bearing spacer |
| 9 | Bearing 608zz |
| 10 | Bearing enclosure |
| 11 | Bearing spacer |
| 12 | Bearing 608zz |
| 13 | Bearing enclosure |
| 14 | Bearing spacer |
| 15 | Bearing 608zz |
| 16 | Top bearing enclosure |
| 17 | Lock nut |
| 18 | Top platform |
| 19 | Knee wire guide |
| 20 | Compass holder |
| 21 | LoRa holder |

Part List

Рисунок Б.3 — Розбірна модель пристрою турелі

Продовження додатку Б

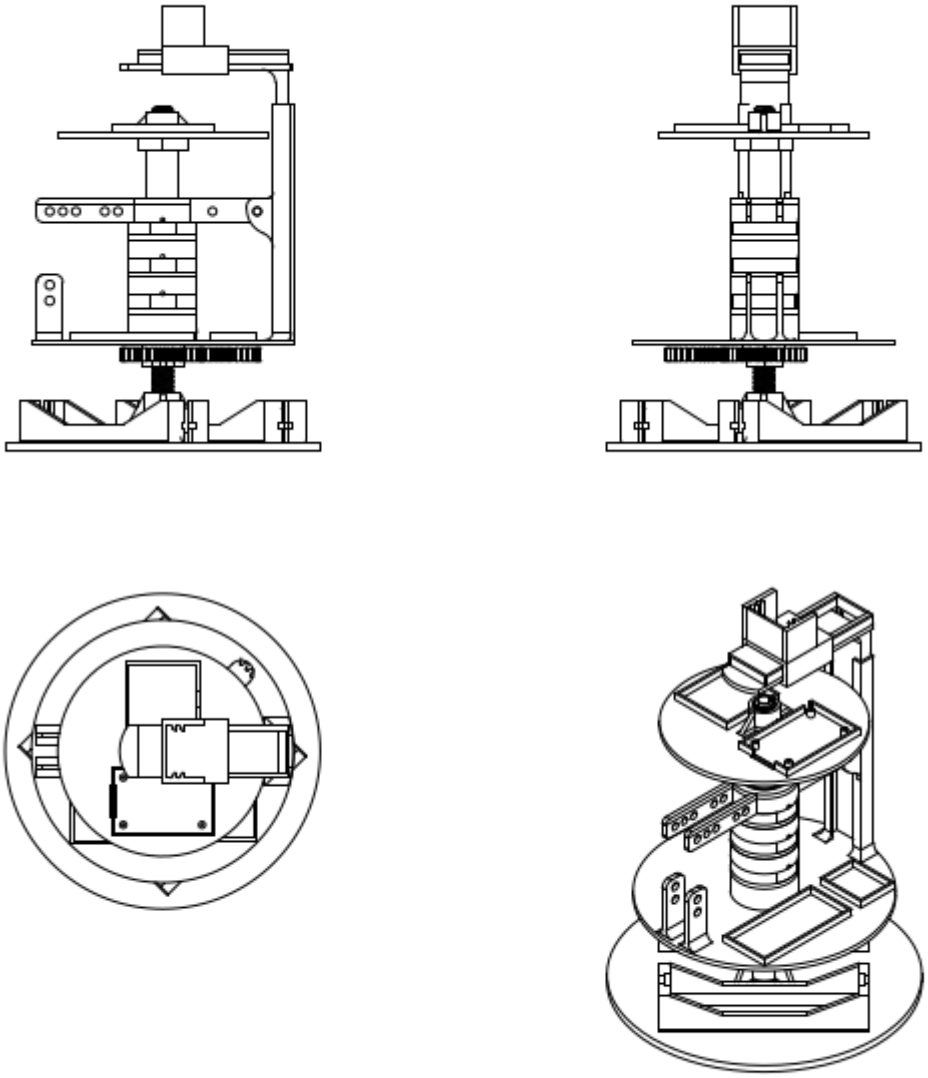


Рисунок Б.4 — Збірна модель пристрою турелі

Продовження додатку Б

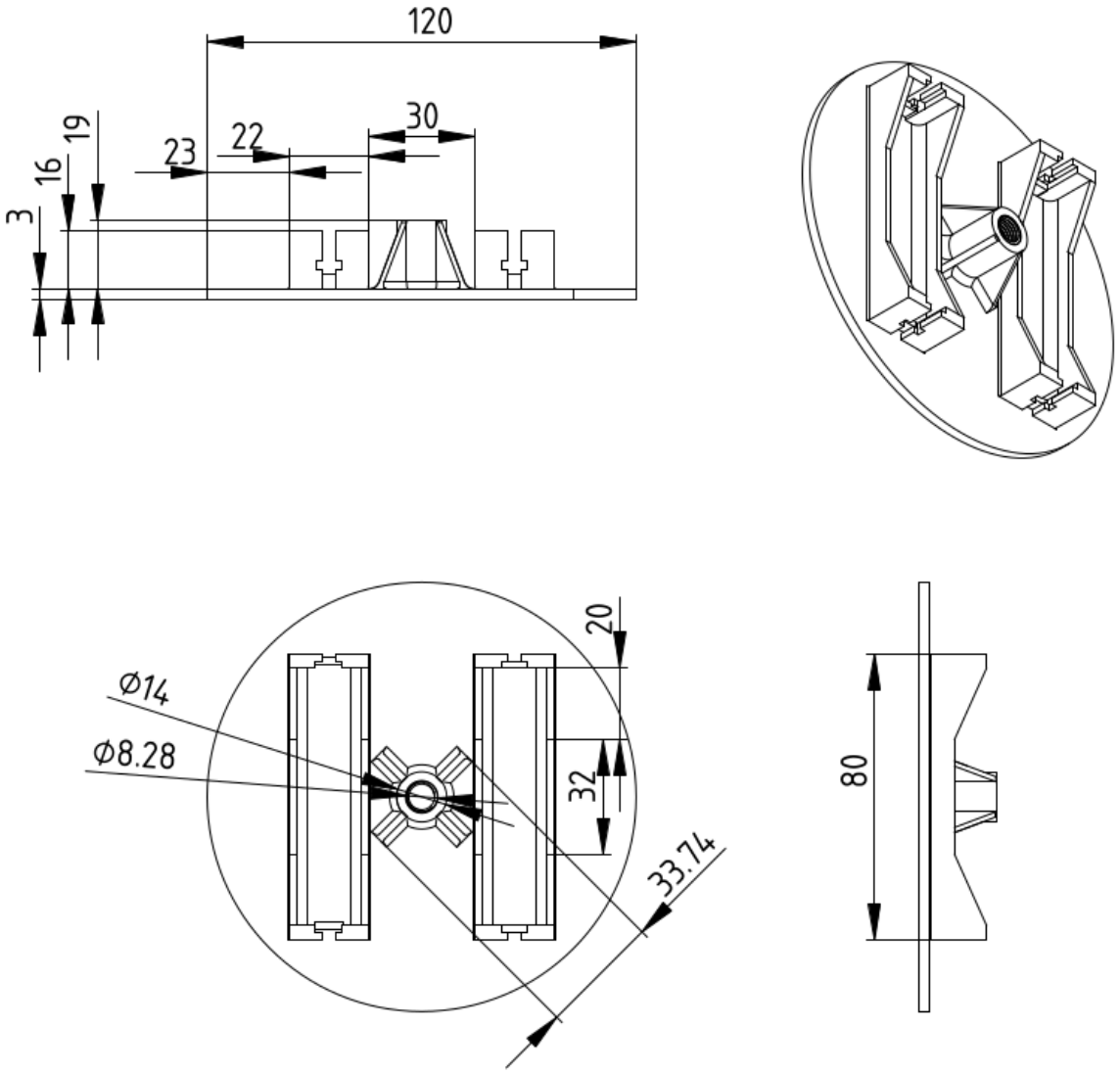



Рисунок Б.5 — Нижня платформа статичної частини турелі

Продовження додатку Б

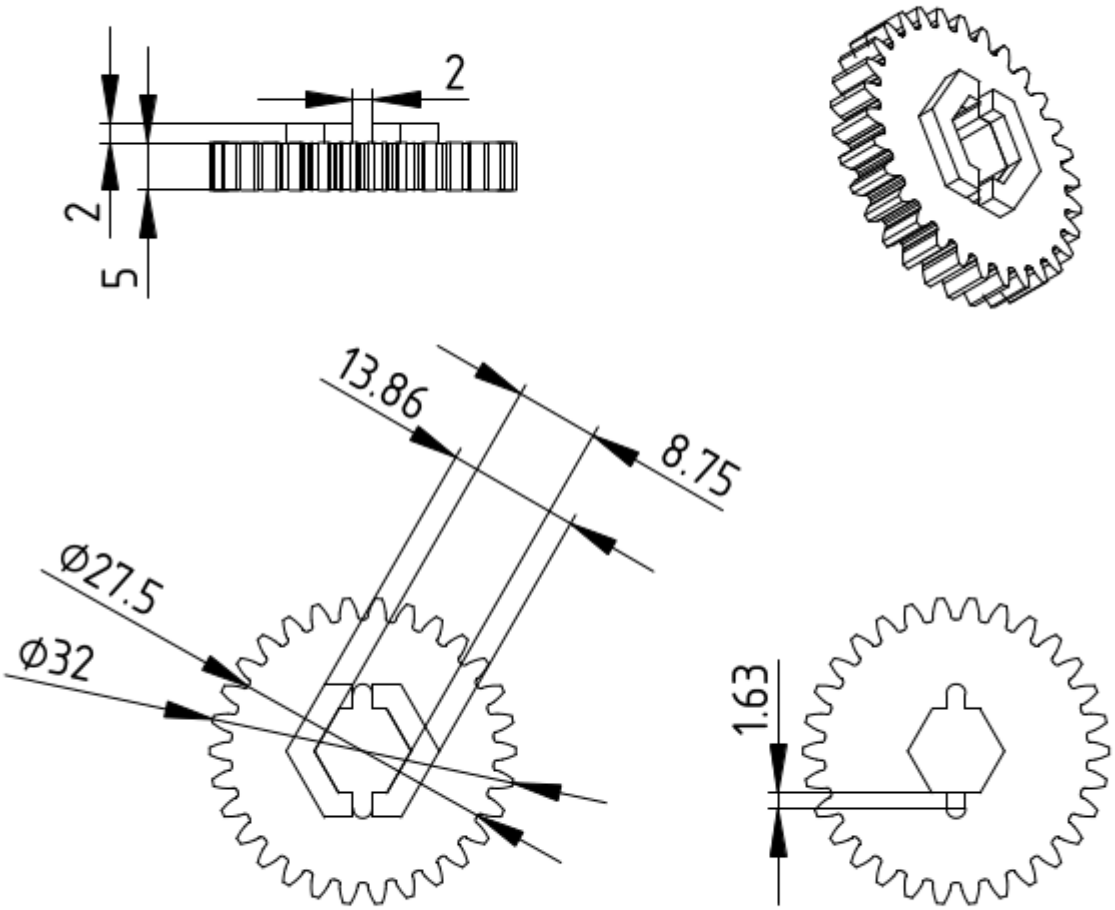



Рисунок Б.6 — Опорна шестерня

Продовження додатку Б

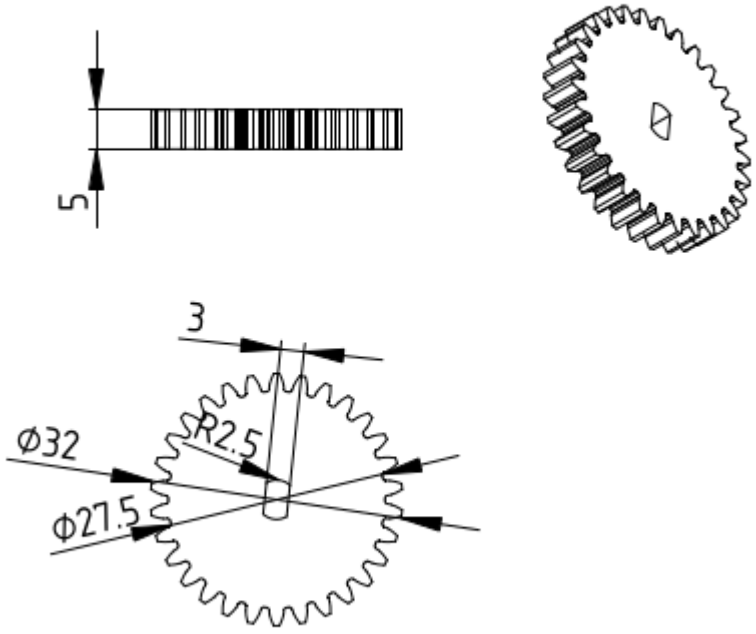



Рисунок Б.7 — Шестерня крокового двигуна

Продовження додатку Б

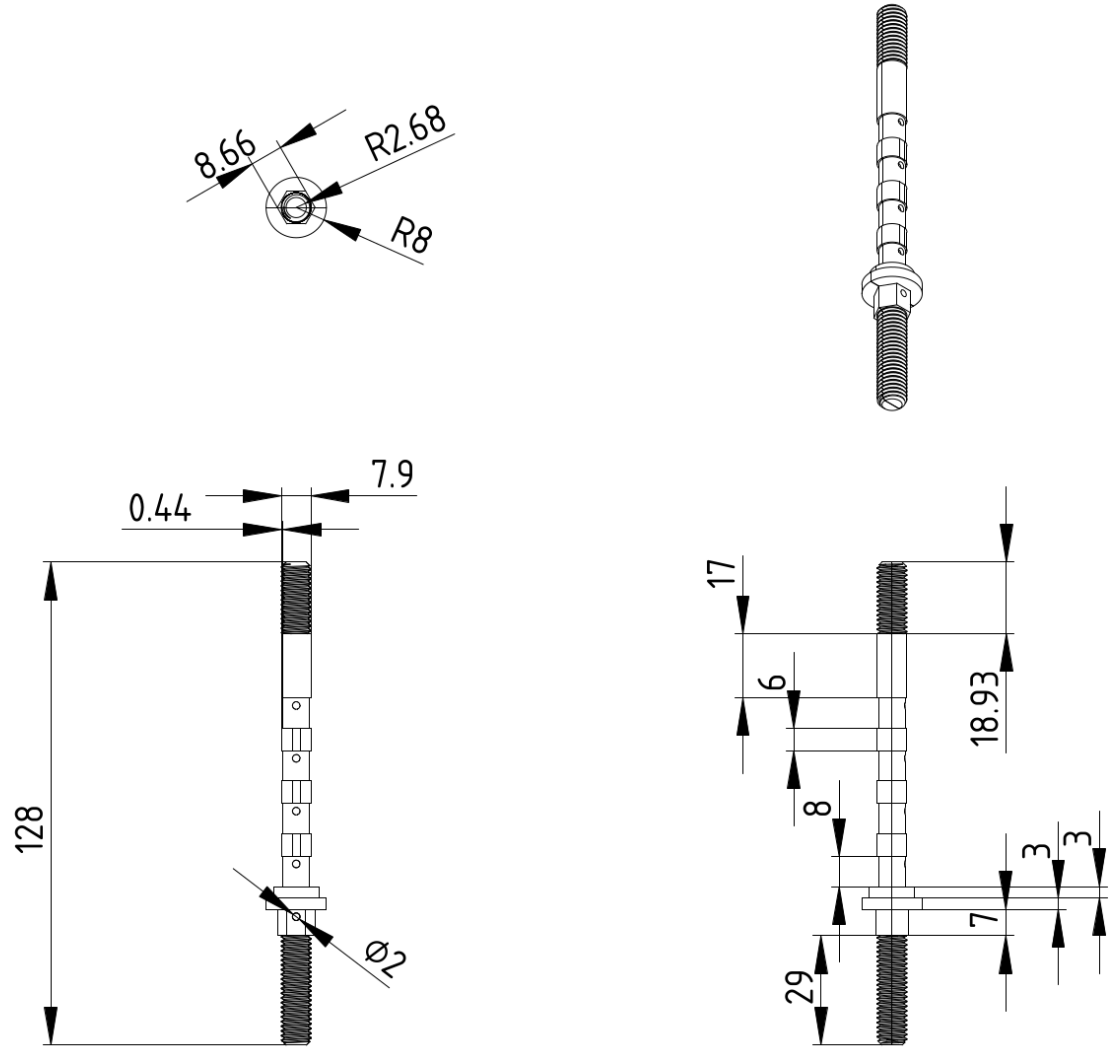



Рисунок Б.8 — Пін вісь

Продовження додатку Б

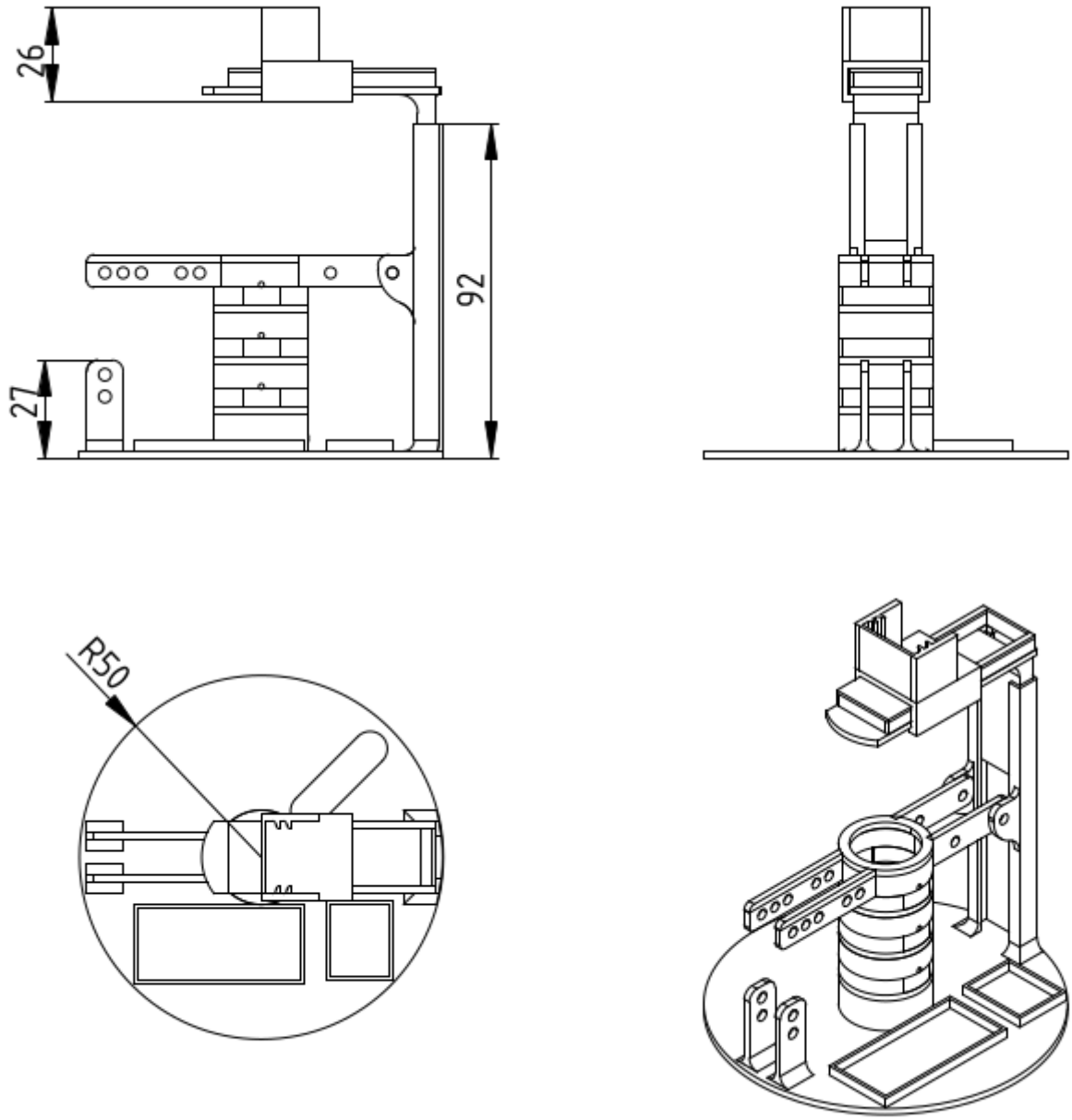



Рисунок Б.9 — Збірна модель обертової частини турелі

Продовження додатку Б

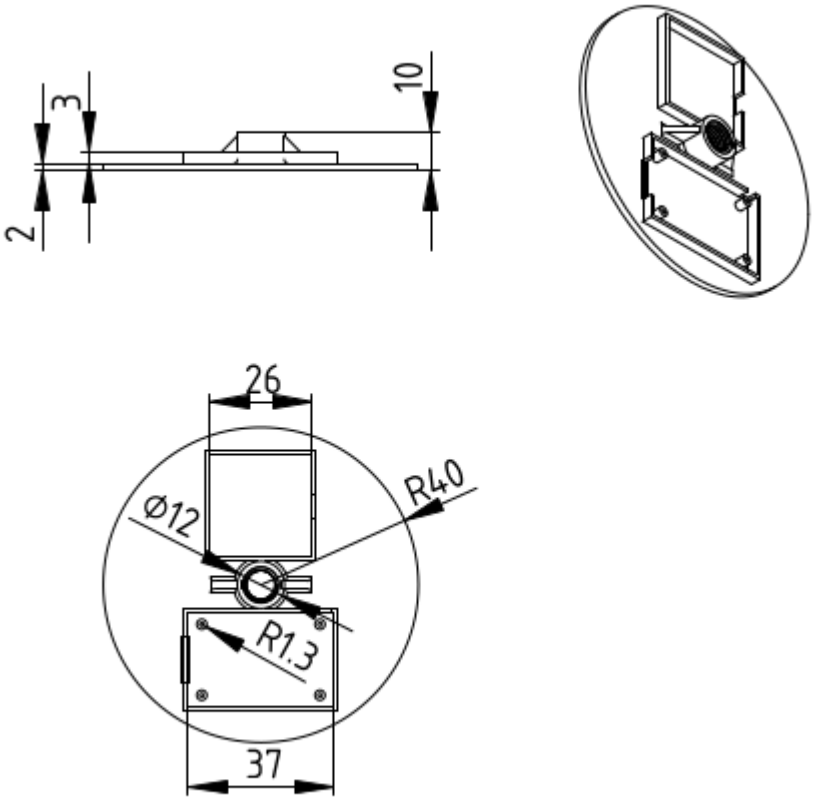



Рисунок Б.10 — Верхня GPS платформа

Продовження додатку Б

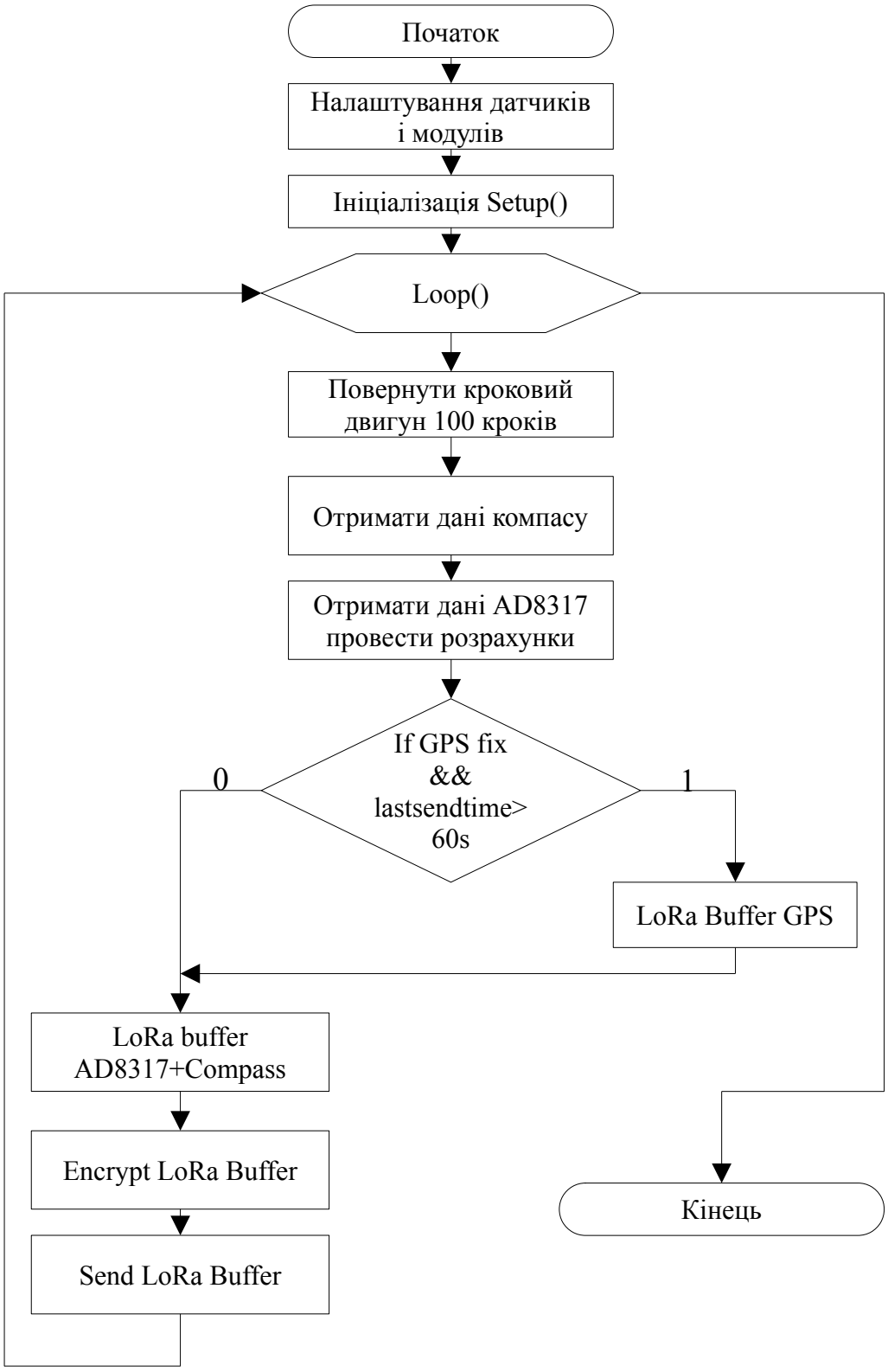



Рисунок Б.11 — Алгоритм функціонування пристрою турелі

# Додаток В

## (обов'язковий)

## Лістинг програм

```
#програма управління туреллю
#include <SPI.h>
#include <Wire.h>
#include <QMC5883LCompass.h>
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include <LoRa.h>
#include <Stepper.h>
#include <Crypto.h>
#include <AES.h>
#include <string.h>
//#include <Base64.h>

// Define step constant
#define STEPS 100

// Creates an instance
// Pins entered in sequence IN1-IN3-IN2-IN4 for proper step sequence
Stepper stepper(STEPS, 5, 7, 6, 10);


typedef struct RFDATA
{
int avg_rawADC,max_rawADC,min_rawADC;
int low_f,max_f,mid_f;
float min_V,max_V,avg_V;
// float min_W,max_W,avg_W;
float min_dbm,max_dbm,avg_dbm;
} RFDATA;

typedef struct COMPASSDATA
{
int x, y, z, a, b;
char compass_direction[3];
} COMPASSDATA;

RFDATA adc8317_0;
COMPASSDATA compassdata;

#define LORA_BUFFER_LIMIT 210

char lora_buffer[LORA_BUFFER_LIMIT]={'\0'};
//char lora_buffer_decrypted[LORA_BUFFER_LIMIT]={'\0'};
char floatString_buffer1[10]={'\0'};
char floatString_buffer2[10]={'\0'};
char floatString_buffer3[10]={'\0'};

static const byte lora_trgtIdA PROGMEM=0x0B;
static const byte lora_trgtIdB PROGMEM=0x80;
static const byte lora_devIdA PROGMEM=0xAA;
static const byte lora_devIdB PROGMEM=0xAA;
unsigned char lora_batchId = 20;
unsigned char lora_msgCnt = 20;

static const byte key[16] PROGMEM ={0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07,0x08,
0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F};
char cipher_buffer[16] = {'\0'};
AESTiny128 aestiny128;
```

Продовження додатку В

```
#define GPS_BASE_DATA_JSON "age:%"PRIu32",isValid:%u,"
#define COMPAS_DATA_JSON_TMP "compass:{x:%d,y:%d,z:%d,a:%d,b:%d,d:\"%c%c%c\"\}"
#define RF_DATA_JSON_TMP "rf:{gdb:%s,ndb:%s,xdb:%s\}"


static const char COMPAS_DATA_JSON[] PROGMEM = "compass:{x:%d,y:%d,z:%d,a:%d,b:%d,d:\"%c
%c%c\"\}";
static const char GPS_SAT_DATA_JSON[] PROGMEM = "sat:{"GPS_BASE_DATA_JSON"value:
%"PRIu32"\}";
//10th of degrees means divide by 100 to get degrees
static const char GPS_COURSE_DATA_JSON[] PROGMEM = "course:{"GPS_BASE_DATA_JSON"value:
%"PRIu32"\}";
// non parsed value stands for ddMMyy
static const char GPS_DATE_DATA_JSON[] PROGMEM = "date:{"GPS_BASE_DATA_JSON"value:
%"PRIu32"\}";
//for saving space just send non parsed value data format is HHmmss (cs0=zzz
1centisecond=10ms)
static const char GPS_TIME_DATA_JSON[] PROGMEM = "time:{"GPS_BASE_DATA_JSON"value:
%"PRIu32"\}";
//hdop appears to be in centimeters so 100 means 1,150->1.5 and so on
static const char GPS_HDOP_DATA_JSON[] PROGMEM = "hdop:{"GPS_BASE_DATA_JSON"value:
%"PRIu32"\}";
//value 10th of knots example if vale 43 then 43*1.852/100
static const char GPS_SPEED_DATA_JSON[] PROGMEM = "speed:{"GPS_BASE_DATA_JSON"value:
%"PRIu32"\}";
static const char GPS_LOCATION_DATA_JSON[] PROGMEM = "loc:{"GPS_BASE_DATA_JSON"lat:
%s,lng:%s\}";
//altitude value is in centimeters
static const char GPS_ALTITUDE_DATA_JSON[] PROGMEM = "alt:{"GPS_BASE_DATA_JSON"value:
%"PRIu32"\}";
static const char RF_DATA_JSON[] PROGMEM = RF_DATA_JSON_TMP;
static const char COMPASS_RF_DATA_JSON[] PROGMEM =
COMPAS_DATA_JSON_TMP","RF_DATA_JSON_TMP;

//AD8317
const float AD_RESOLUTION = 1024.0; // AD resolution - 1 = 2^10 - 1

#define aref_voltage 2.048
int ADCReading;
#define AD8317_INPUT A0
#define NUM_SAMPLES 20
unsigned char sample_count =0; // current sample number

long sum =0; // sum of samples taken
int min=1000;
int max=-1000;
int tempread=0;
int dbm_at_0V=7; //calibrated value better to have array for different rf ranges
const float MV_DB_SLOPE = 0.022;

//GPS
static const int RXPin = 2; int TXPin = 3; //software serial

QMC5883LCompass compass;
SoftwareSerial gpsSerial(RXPin, TXPin);
unsigned long last = 0UL;
TinyGPSPlus gps;
//Adafruit_BMP280 bmp; // use I2C interface

#define SEALEVELPRESSURE_HPA (1013.25)

//Lora
#define ss 9
#define rst 8
```

Продовження додатку В

```
#define dio0 4


void setup() {
aestiny128.setKey(key,16);
analogReference(EXTERNAL);
Serial.begin(9600);

while (!Serial);
Serial.println("Project69");

Wire.begin();

compass.init();
compass.setSmoothing(3, true);

gpsSerial.begin(9600); //bad way

LoRa.setPins(ss, rst, dio0);
if (!LoRa.begin(433E6)) {
Serial.println("Starting LoRa failed!");
while (1);
}
//LoRa.enableCrc();
stepper.setSpeed(60);
}

void loop() {

Serial.println(F("Move motor"));
stepper.step(100);
ReadCompass();
ReadAD8317();
dtostrf(adc8317_0.avg_dbm,4,2,floatString_buffer1);
dtostrf(adc8317_0.min_dbm,4,2,floatString_buffer2);
dtostrf(adc8317_0.max_dbm,4,2,floatString_buffer3);
sprintf_P((char
*)lora_buffer,COMPASS_RF_DATA_JSON,compassdata.x,compassdata.y,compassdata.z,compassdata.
a
,compassdata.b
,compassdata.compass_direction[0]
,compassdata.compass_direction[1]
,compassdata.compass_direction[2]
,floatString_buffer1
,floatString_buffer2
,floatString_buffer3);
LoraSend();
floatString_buffer1[0]='\0';
floatString_buffer2[0]='\0';
floatString_buffer3[0]='\0';

while (gpsSerial.available() > 0)
{
if (gps.encode(gpsSerial.read())) {
GPSToSerial();
}
}
if((unsigned int)lora_batchId<255)
{
++lora_batchId;
}
else
{
lora_batchId=20;
}
```

Продовження додатку В

```
}
int encryptLoraBuffer(char* lora_buffer,char* buffer)//,int buffer_size)

{
unsigned long start;
unsigned long elapsed;
start = micros();
int buffer_size=strlen(lora_buffer);
int offset = 0;
int block_size = 16;
int cnt=0;
while (offset < buffer_size) {
Serial.println(cnt);
memset(buffer, 0, sizeof(buffer));
//int block_size = min(buffer_size - offset, 16);
// Copy 16 bytes from lora_buffer to cipher_buffer
memcpy(buffer, lora_buffer + offset,block_size);
Serial.print("enc:");
Serial.println(buffer);
// Encrypt the block in cipher_buffer
aestiny128.encryptBlock(buffer,buffer);//cypher->output block and plaintext->input block
Serial.print("enc:");
Serial.println(buffer);
// Replace the original data in lora_buffer with the encrypted data
memcpy(lora_buffer + offset, buffer, block_size);
offset += block_size; // Move to the next 16-byte block
cnt++;
}
elapsed = micros() - start;
Serial.print(elapsed / ((float)buffer_size * 16.0));
Serial.println("us per byte, ");
return offset;
}

void LoraSend()
{
Serial.print("b:");
Serial.print((unsigned int)lora_batchId,DEC);
Serial.print("; c:");
Serial.println((unsigned int)lora_msgCnt,DEC);
Serial.print("Before Encryption(");
Serial.print(strlen(lora_buffer));
Serial.print("):");
Serial.println(lora_buffer);
//enable encryption
int encryptedsize=encryptLoraBuffer(lora_buffer,cipher_buffer);//,sizeof(lora_buffer));
int encodedLength = Base64.encodedLength(encryptedsize);
int encodedLength = Base64.encodedLength(strlen(lora_buffer));

//Base64.encode(lora_buffer, lora_buffer, encryptedsize);

//check for success and analyze
LoRa.beginPacket();
LoRa.write(lora_trgtIdA);
LoRa.write(lora_trgtIdB);
LoRa.write(lora_devIdA);
LoRa.write(lora_devIdB);
LoRa.write(lora_batchId);
LoRa.write(lora_msgCnt);
LoRa.print(lora_buffer);
LoRa.endPacket();
memset(lora_buffer, 0, sizeof(lora_buffer));
lora_buffer[0]='\0';
if((unsigned int)lora_msgCnt<255)
{
```

Продовження додатку В

```
++lora_msgCnt;
}
else


{
lora_msgCnt=20;
}
}

void ReadCompass()
{
// Read compass values
compass.read();

compassdata.x = compass.getX();
compassdata.y = compass.getY();
compassdata.z = compass.getZ();
compassdata.a = compass.getAzimuth();

compassdata.b = compass.getBearing(compassdata.a);
compass.getDirection(compassdata.compass_direction, compassdata.a);

}


void GPSToSerial()
{
if (millis() - last > 60000)
{
sprintf_P((char *)lora_buffer,
GPS_SAT_DATA_JSON,gps.satellites.age(),gps.satellites.isValid(),gps.satellites.value());
LoraSend();
sprintf_P((char *)lora_buffer,
GPS_COURSE_DATA_JSON,gps.course.age(),gps.course.isValid(),gps.course.value());
LoraSend();

sprintf_P((char *)lora_buffer,
GPS_DATE_DATA_JSON,gps.date.age(),gps.date.isValid(),gps.date.value());
LoraSend();

sprintf_P((char *)lora_buffer,
GPS_TIME_DATA_JSON,gps.time.age(),gps.time.isValid(),gps.time.value());
LoraSend();

sprintf_P((char *)lora_buffer,
GPS_HDOP_DATA_JSON,gps.hdop.age(),gps.hdop.isValid(),gps.hdop.value());
LoraSend();

sprintf_P((char *)lora_buffer,
GPS_SPEED_DATA_JSON,gps.speed.age(),gps.speed.isValid(),gps.speed.value());
LoraSend();

dtostrf(gps.location.lat(),4,5,floatString_buffer1);
dtostrf(gps.location.lng(),4,5,floatString_buffer2);
sprintf_P((char *)lora_buffer,
GPS_LOCATION_DATA_JSON,gps.location.age(),gps.location.isValid()
,floatString_buffer1
,floatString_buffer2);
LoraSend();
floatString_buffer1[0]='\0';
floatString_buffer2[0]='\0';

sprintf_P((char *)lora_buffer,
GPS_ALTITUDE_DATA_JSON,gps.altitude.age(),gps.altitude.isValid(),gps.altitude.value());
```

Продовження додатку В

```
LoraSend();

Serial.print(F("DIAGS Chars="));
Serial.print(gps.charsProcessed());

Serial.print(F(" Sentences-with-Fix="));
Serial.print(gps.sentencesWithFix());
Serial.print(F(" Failed-checksum="));
Serial.print(gps.failedChecksum());
Serial.print(F(" Passed-checksum="));
Serial.println(gps.passedChecksum());

if (gps.charsProcessed() < 10)
Serial.println(F("WARNING: No GPS data. Check wiring."));
last = millis();
}

}

void ReadAD8317()
{
tempread=0;
ADCReading=0;
sample_count=0;
min=-1000;
max=1000;
sum=0;
tempread=0;
while (sample_count < NUM_SAMPLES) {
tempread= analogRead(AD8317_INPUT);
sum += tempread;
if(tempread > min){min=tempread;}
if(tempread < max){max=tempread;}

sample_count++;

}

ADCReading=sum/NUM_SAMPLES;
adc8317_0.avg_rawADC=ADCReading;
adc8317_0.min_rawADC=min;
adc8317_0.max_rawADC=max;

adc8317_0.avg_V = ADCReading * aref_voltage;
adc8317_0.avg_V /= AD_RESOLUTION;

adc8317_0.min_V = min * aref_voltage;
adc8317_0.min_V /= AD_RESOLUTION;

adc8317_0.max_V = max * aref_voltage;
adc8317_0.max_V /= AD_RESOLUTION;

adc8317_0.avg_dbm =convertVoltageToDbm(adc8317_0.avg_V);
adc8317_0.min_dbm =convertVoltageToDbm(adc8317_0.min_V);
adc8317_0.max_dbm =convertVoltageToDbm(adc8317_0.max_V);

}

float convertVoltageToDbm(float voltage)
{
return dbm_at_0V -(voltage/MV_DB_SLOPE);
}
```

Продовження додатку В

```
#програма управління пристроєм приймачем
#include <SPI.h>
#include <LoRa.h>
#include <Crypto.h>
#include <AES.h>
#include <string.h>
#include <Base64.h>

//Lora
#define ss 9
#define rst 8
#define dio0 4

#define MAX_PACKET_SIZE 256
char receivedData[MAX_PACKET_SIZE]={'\0'};
char packetid_buffer[6]={'\0'};
char lora_buffer[216]={'\0'};
static const byte key[16] PROGMEM ={0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07,0x08,
0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F};
char cipher_buffer[16] = {'\0'};
AES128 aes128;

void decryptLoraBuffer(char* lora_buffer,char* buffer,int buffer_size)
{
     unsigned long start;
     unsigned long elapsed;
     start = micros();
     //int buffer_size=strlen(lora_buffer);
     int block_size = 16;
     int offset = 0;
     while (offset < buffer_size) {
          memset(buffer, 0, sizeof(buffer));
          //int block_size = min(buffer_size - offset, 16);
          memcpy(buffer, lora_buffer + offset, block_size);
          aes128.decryptBlock(buffer,buffer);
          memcpy(lora_buffer + offset, buffer, block_size);
          offset += block_size; // Move to the next 16-byte block
     }
   elapsed = micros() - start;
   Serial.print(elapsed / ((float)buffer_size * 16.0));
   Serial.println("us per byte, ");
}

void setup() {
   aes128.setKey(key,16);
   Serial.begin(9600);
   while (!Serial);

   Serial.println("LoRaReceiver");
   LoRa.setPins(ss, rst, dio0);
   if (!LoRa.begin(433E6)) {
      Serial.println("Starting LoRa failed!");
      while (1);
   }
   //LoRa.enableCrc();
}

void loop() {
   // try to parse packet

   int packetSize = LoRa.parsePacket();

  if (packetSize) {
     // received a packet

     memset(receivedData, 0, MAX_PACKET_SIZE); // Initialize the buffer with zeros
```

Продовження додатку В

```
     memset(packetid_buffer, 0, 6);

     int bytesRead = 0; // Keep track of the number of bytes read

     // read packet
     int cnt=0;
     while (LoRa.available()) {
          if (bytesRead < 6) {
          // Read the first 6 bytes into the firstSixBytes buffer
          packetid_buffer[bytesRead] = (char)LoRa.read();
        } else {
          // Read the remaining data into the remainingData buffer
          receivedData[bytesRead - 6] = (char)LoRa.read();
        }

        bytesRead++;

        // Check if we've reached the maximum packet size
        if (bytesRead >= MAX_PACKET_SIZE) {
          break;
        }
          cnt++;
     }
     decryptLoraBuffer(receivedData,cipher_buffer,bytesRead-6);
     char joinedData[MAX_PACKET_SIZE];
     memcpy(joinedData, packetid_buffer, 6);
     memcpy(joinedData + 6, receivedData, MAX_PACKET_SIZE - 6);
     Serial.print(joinedData);
     Serial.print("|Lorapacket");
     // print RSSI of packet
     Serial.print("|RSSI:");
     Serial.println(LoRa.packetRssi());

   }
}
```

**#Програмне забезпечення візуалізації**

loramessageparser.h

```
#ifndef LORAMESSAGEPARSER_H
#define LORAMESSAGEPARSER_H

#include <QObject>
#include <QWidget>
#include <QRegularExpression>
#include <QRegularExpressionMatchIterator>
#include <QDebug>
#include <QJsonDocument>
#include <QJsonValue>
#include <QJsonArray>
#include <QJsonObject>

class LoraMessageParser: public QObject
{
    Q_OBJECT
public:
    explicit LoraMessageParser(QObject *parent = nullptr);
    ~LoraMessageParser();
    QString loraMessage;
signals:
    void loraNewParsedMeasage(QString loraMessage);

public slots:
```

Продовження додатку В

```
    void loraMessageArrived(QByteArray header,QString loraMessage);

};

#endif // LORAMESSAGEPARSER_H
```

mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include<QtSerialPort>
#include<QSerialPortInfo>
#include "serialportreader.h"
#include "loramessageparser.h"
#include <QStandardItemModel>
#include <QScrollBar>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

enum historyColumns {
    hisoryTimeColumnID,
    historyMessageColumnID
};

class MainWindow : public QMainWindow

{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();
    QStandardItemModel *history_model;

private:
    Ui::MainWindow *ui;

    SerialPortReader *serialPortReader;
    LoraMessageParser *loraMessageParser;
    void updateDeviceList();

private slots:
    void ondevice_comboBox_currentIndexChanged();

    void on_connect_pushButton_clicked();
    void updateText(QString text);
    void on_refresh_toolButton_clicked();

    void on_history_model_rowsInserted(const QModelIndex & parent, int start, int end);
    void on_history_tableView_selectionChanged(const QItemSelection&, const
QItemSelection&);
};

#endif // MAINWINDOW_H
```

Продовження додатку В

serialportreader.h

```
#ifndef SERIALPORTREADER_H

#define SERIALPORTREADER_H

#include <QSerialPort>
#include <QObject>
#include <QWidget>
#include <openssl/aes.h>

static const unsigned char key[16] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07,
0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F};
static const QString appID="b80";

class SerialPortReader : public QObject
{
    Q_OBJECT
    Q_PROPERTY(QString portName
               READ getportName
               WRITE setportName
              )
    Q_PROPERTY(int baudRate
               READ getbaudRate
               WRITE setbaudRate
              )
public:
    explicit SerialPortReader(QObject *parent = nullptr);
    ~SerialPortReader();


    void setportName(QString m_portName)

    {
        portName = m_portName;
        //emit portName_changed();
    }
    QString getportName() const
    {
        return portName;
    }

    void setbaudRate(int m_baudRate)
    {
        baudRate = m_baudRate;
        //emit baudRate_changed();
    }
    int getbaudRate() const
    {
        return baudRate;
    }
    AES_KEY aesKey;
    unsigned char cipherText[16];

public slots:
    void startPort();
    void stopPort();

private:
    QString portName;
    int baudRate;
```

```
    QSerialPort *m_serialPort = nullptr;
    QByteArray m_readData;
    QByteArray key16;

signals:
    void serialPortNewData(QString line);
    void serialLoRaAppMessage(QByteArray header, QString line);
    void serialLoRaUnknownMessage(QString line);
    void portName_changed();
    void baudRate_changed();
    void serialPortErrorSignal(QString line);

private slots:
    void onPortName_changed();
    void onBaudRate_changed();
    void onPort_started();
    void onPort_stopped();
    void readData();
    void serialError(QSerialPort::SerialPortError serialPortError);
};

#endif // SERIALPORTREADER_H
```

loramessageparser.cpp

```
#include "loramessageparser.h"

LoraMessageParser::LoraMessageParser(QObject *parent)
    : QObject{parent}
{

}


LoraMessageParser::~LoraMessageParser()
{

}

void LoraMessageParser::loraMessageArrived(QByteArray header, QString loraMessage)
{

    QJsonArray jsonArray;

    // Process the extracted data as needed
    qDebug() << "Extracted Data:" << loraMessage;

    for (int i = 0; i < header.size(); ++i)
        {
            quint8 byteValue = static_cast<quint8>(header.at(i)); // Extract the byte
value
            jsonArray.append(static_cast<int>(byteValue)); // Append the byte value to
the JSON array
        }

    QRegExp regex("(\\w+):");

    // Replace all matches with the word enclosed in double quotes
    loraMessage.replace(regex, "\"\\1\":");
```

Продовження додатку В

```
    qDebug()<<loraMessage;

    QJsonDocument loraJsonDocument
=QJsonDocument::fromJson(QString("{"+loraMessage+"}").toUtf8());
    qDebug()<<loraJsonDocument.toJson(QJsonDocument::Compact);
    QJsonObject loraJsonObject = loraJsonDocument.object();
    qDebug()<<loraJsonObject;
    loraJsonObject.insert("header", jsonArray);
    //loraJsonObject.insert("data", loraJsonObject);
    QJsonDocument resultDocument(loraJsonObject);
    QString loraMessagePreparsed = resultDocument.toJson(QJsonDocument::Compact);
    qDebug() << loraMessagePreparsed;

    emit loraNewParsedMeasage(loraMessagePreparsed);
    //QRegularExpression reg("Lorapacket:'(.+})'RSSI:([+-]?\\d+)");

}
```

serialportreader.cpp

```
#include "serialportreader.h"
#include "qserialportinfo.h"
#include <QDebug>

SerialPortReader::SerialPortReader(QObject *parent) : QObject{parent}
{

    AES_set_decrypt_key(key, 128, &aesKey);

    m_serialPort = new QSerialPort();
    connect(m_serialPort, &QSerialPort::readyRead, this,
            &SerialPortReader::readData);

     connect(m_serialPort, &QSerialPort::errorOccurred, this,

     &SerialPortReader::serialError);
}

SerialPortReader::~SerialPortReader()

{
    stopPort();
    m_serialPort->deleteLater();
}

void SerialPortReader::onPortName_changed() {}

void SerialPortReader::onBaudRate_changed() {}

void SerialPortReader::startPort()
{
    qDebug() << "startPort()" << getportName() << getbaudRate();
    stopPort();
    m_serialPort->setPortName(getportName());
    m_serialPort->setBaudRate(getbaudRate());
    if (!m_serialPort->open(QIODevice::ReadOnly))
        {
            qDebug() << "failed to open";
        }
}
```

Продовження додатку В

```
void SerialPortReader::stopPort()
{
    if (m_serialPort->isOpen())
        {
            qDebug() << "close current port";
            m_serialPort->close();
        }
}

void SerialPortReader::onPort_started() {}

void SerialPortReader::onPort_stopped() {}

void SerialPortReader::readData()
{
    qDebug() << "readData()";
    while (m_serialPort->canReadLine())
        {
            QByteArray line = m_serialPort->readLine();
            if (line.toLower().contains("lorapacket"))
                {
                    QString messageAppId = QString::number(static_cast<quint8>(line[0]),
16) +
                                           QString::number(static_cast<quint8>(line[1]),
16);
                    QByteArray header=line.left(6);
                    line.remove(0, 6);
                    qDebug() << line;
                    int delimiterPos = line.indexOf("|Lorapacket");
                    QByteArray extractedData = "";
                    if (delimiterPos != -1)
                        {
                            // Extract the data up to the delimiter
                            extractedData = line.left(delimiterPos);

                            if (messageAppId == appID)
                                {

                                    emit
serialLoRaAppMessage(header,QString(extractedData));
                                }
                            else
                                {
                                    emit serialLoRaUnknownMessage(QString(line));

                                }
                        }
                    else
                        {
                            emit serialPortNewData(QString(line));
                        }
                }
        }
}

void SerialPortReader::serialError(
    QSerialPort::SerialPortError serialPortError)
{
    qDebug() << "serialError";
    if (serialPortError == QSerialPort::ReadError)
        {
            emit serialPortErrorSignal("Error " + m_serialPort->portName() + " " +
                                       m_serialPort->errorString());
```

```
        }
}
```

main.cpp

```
#include "mainwindow.h"
#include <QApplication>
#include <QtDebug>
#include <QDateTime>
#include<QLoggingCategory>

void qtLogger(QtMsgType type, const QMessageLogContext &context, const QString &msg)
{
    QByteArray localMsg = msg.toLocal8Bit();
    const char *file = context.file ? context.file : "";
    const char *function = context.function ? context.function : "";
    const char *timestamp =
QDateTime::currentDateTime().toString(Qt::DateFormat::ISODate).toUtf8();
    const char *level = "n/a";

    switch (type)
        {
        case QtDebugMsg:
            level = "DEBUG";
            break;
        case QtInfoMsg:
            level = "INFO";
            break;
        case QtWarningMsg:
            level = "WARNING";
            break;
        case QtCriticalMsg:
            level = "CRITICAL";
            break;
        case QtFatalMsg:
            level = "FATAL";

            break;
        }


    fprintf(stderr, "%s [%s] %s (%s:%u, %s)\n",
            timestamp, level, localMsg.constData(), file, context.line, function);
}

int main(int argc, char *argv[])
{
    qInstallMessageHandler(qtLogger);

    QApplication a(argc, argv);

    QLoggingCategory::setFilterRules("*.debug=true\nqt.*.debug=false");
    MainWindow w;
    w.show();
    return a.exec();
}
```

mainwindow.cpp

```
#include "mainwindow.h"
#include <QDebug>
#include "qstringliteral.h"
#include "ui_mainwindow.h"
```

Продовження додатку В

```
#include "MapGraphicsView.h"
#include "MapGraphicsScene.h"
#include "tileSources/GridTileSource.h"
#include "tileSources/OSMTileSource.h"
#include "tileSources/CompositeTileSource.h"
#include "guts/CompositeTileSourceConfigurationWidget.h"
#include "CircleObject.h"
#include "PolygonObject.h"

#include <QSharedPointer>
#include <QtDebug>
#include <QThread>
#include <QImage>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    this->setWindowState(Qt::WindowMaximized);
    //Setup the MapGraphics scene and view
    MapGraphicsScene * scene = new MapGraphicsScene(this);
    MapGraphicsView * view = new MapGraphicsView(scene,this);


    view->setParent(ui->mapwidget);

    //Setup some tile sources
    QSharedPointer<OSMTileSource> osmTiles(new OSMTileSource(OSMTileSource::OSMTiles),
&QObject::deleteLater);
    QSharedPointer<GridTileSource> gridTiles(new GridTileSource(),
&QObject::deleteLater);

    QSharedPointer<CompositeTileSource> composite(new CompositeTileSource(),
&QObject::deleteLater);
    composite->addSourceBottom(osmTiles);
    composite->addSourceTop(gridTiles);
    view->setTileSource(composite);

    //Create a widget in the dock that lets us configure tile source layers
    CompositeTileSourceConfigurationWidget * tileConfigWidget = new
CompositeTileSourceConfigurationWidget(composite.toWeakRef(),

this->ui->dockWidget);
    this->ui->dockWidget->setWidget(tileConfigWidget);
    delete this->ui->dockWidgetContents;


    view->setZoomLevel(12);
    view->centerOn(28.471115, 49.22815);

    // Create a circle on the map to demonstrate MapGraphicsObject a bit
    // The circle can be clicked/dragged around and should be ~5km in radius
    MapGraphicsObject * circle = new CircleObject(5000, false, QColor(255, 0, 0, 100));
    circle->setLatitude(49.22815);
    circle->setLongitude( 28.471115);
    scene->addObject(circle);

    updateDeviceList();
    serialPortReader= new SerialPortReader();
    loraMessageParser= new LoraMessageParser();
    connect(ui->device_comboBox, &QComboBox::currentTextChanged, this,
&MainWindow::ondevice_comboBox_currentIndexChanged);
```

Продовження додатку В

```
connect(serialPortReader,&SerialPortReader::serialLoRaAppMessage,loraMessageParser,&LoraM
essageParser::loraMessageArrived);

connect(loraMessageParser,&LoraMessageParser::loraNewParsedMeasage,this,&MainWindow::upda
teText);


    history_model=new QStandardItemModel(0,0,this);
    history_model->setHorizontalHeaderItem(hisoryTimeColumnID,new
QStandardItem(QString("Time")));
    history_model->setHorizontalHeaderItem(historyMessageColumnID,new
QStandardItem(QString("Message")));

    ui->history_tableView->setSelectionBehavior(QAbstractItemView::SelectRows);
    ui->history_tableView->setSelectionMode(QAbstractItemView::SingleSelection);
    ui->history_tableView->setModel(history_model);
    ui->history_tableView->setColumnWidth(hisoryTimeColumnID,200);
    ui->history_tableView->setColumnWidth(historyMessageColumnID,500);

    connect(ui->history_tableView->selectionModel(), SIGNAL(selectionChanged(const
QItemSelection&, const
QItemSelection&)),this,SLOT(on_history_tableView_selectionChanged(const QItemSelection&,
const QItemSelection&)));
    connect(ui->history_tableView-
>model(),SIGNAL(rowsInserted(QModelIndex,int,int)),SLOT(on_history_model_rowsInserted(QMo
delIndex,int,int)));

}

MainWindow::~MainWindow()
{
    serialPortReader->deleteLater();

    loraMessageParser->deleteLater();
    history_model->deleteLater();

    delete ui;
}


void MainWindow::ondevice_comboBox_currentIndexChanged()
{
    qDebug()<<ui->device_comboBox->currentData().toString();
}

void MainWindow::on_connect_pushButton_clicked()
{
    qDebug()<<"on_connect_pushButton_clicked()";
    serialPortReader->setportName(ui->device_comboBox->currentData().toString());
    serialPortReader->setbaudRate(9600);


    serialPortReader->startPort();
}

void MainWindow::updateText(QString text)
{
    ui->debug_plainTextEdit->appendPlainText(text+"\n");
    int row=history_model->rowCount();
    history_model->setItem(row,hisoryTimeColumnID,new
QStandardItem(QDateTime::currentDateTime().toString()));
```

Продовження додатку В

```
    history_model->setItem(row,historyMessageColumnID,new QStandardItem(text));
    history_model->setData(history_model-
>index(row,historyMessageColumnID),text,Qt::UserRole);
}

void MainWindow::on_refresh_toolButton_clicked()
{
    updateDeviceList();
}

void MainWindow::updateDeviceList()
{
    ui->device_comboBox->clear();
    const auto infos = QSerialPortInfo::availablePorts();
    for (const QSerialPortInfo &info : infos)
        {
            qDebug()<<info.hasVendorIdentifier()
<<QString::number(info.vendorIdentifier());
            if(info.hasVendorIdentifier() &&
QString::number(info.vendorIdentifier(),16)=="1a86")
                {
                    QString s = tr("Port: ") + info.portName() +
                                tr(" Location: ") + info.systemLocation() +
                                tr("; Description: ") + info.description() +
                                tr("; Manufacturer: ") + info.manufacturer() +
                                tr("; S\\n: ") + info.serialNumber() +
                                tr("; VId: ") + (info.hasVendorIdentifier() ?
QString::number(info.vendorIdentifier(), 16) : QString()) +
                                tr("; PId: ") + (info.hasProductIdentifier() ?
QString::number(info.productIdentifier(), 16) : QString()) +
                                (info.isBusy()?QString(" [Busy]"):QString(""));
                    qInfo()<<info.portName();

                    qInfo()<<s;

                    ui->device_comboBox->addItem(s,info.portName());
                }
        }
}



void MainWindow::on_history_model_rowsInserted(const QModelIndex & parent, int start, int
end)
{
        Q_UNUSED(parent)
        Q_UNUSED(start)
        Q_UNUSED(end)

        if(ui->history_tableView->verticalScrollBar()->value()==ui->history_tableView-
>verticalScrollBar()->maximum())
        {
                ui->history_tableView->scrollToBottom();
        }
}


void MainWindow::on_history_tableView_selectionChanged(const QItemSelection&, const
QItemSelection&)
{
        qDebug()<<"selectionchanged";
}
```

**Додаток Г**

**(обов'язковий)**

**Протокол перевірки магістерської кваліфікаційної роботи на наявність текстових запозичень**

Назва роботи: «Портативна система визначення наявності джерела радіочастотного випромінювання»
Тип роботи: магістерська кваліфікаційна робота
Підрозділ: кафедра АІІТ

**Показники звіту подібності Unicheck**

Оригінальність 98,9 % Схожість 1,1 %

Аналіз звіту подібності (відмітити потрібне)

● Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.
Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і самостійності її автора.
○ Роботу направити на розгляд експертної комісії кафедри.
○ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Роман МАСЛІЙ
(підпис)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Юрій КУЗІН
(підпис)

Керівник роботи _____ Володимир ГАРМАШ