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

Радіотехнічний факультет

Кафедра радіотехнічних систем

До захисту допущено:

Завідувач кафедри

_____ Сергій ЖУК

«___»_____2024 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою

«Радіотехнічні комп'ютеризовані системи»

спеціальності 172 «Телекомунікації та радіотехніка»

на тему: «Апаратне та програмне забезпечення різницево-далекомірної системи на основі Raspberry Pi та SDR»

Виконав:
студент IV курсу, групи РС-01
Панаков Антон Юрійович _____

Керівник:
ст.викл., к.т.н. Товкач Ігор Олегович _____

Рецензент:
ст.викл. Новосад Андрій Анатолійович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Радіотехнічний факультет
Кафедра радіотехнічних систем

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – **172 «Телекомунікації та радіотехніка»**
Освітня програма «**Радіотехнічні комп'ютеризовані системи**»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Сергій ЖУК
«___»_____2024 р.

ЗАВДАННЯ

на дипломну роботу студенту
Прізвище, ім'я, по батькові

1. Тема роботи «Апаратне та програмне забезпечення різницево-далекомірної системи на основі Raspberry Pi та SDR», керівник роботи Товкач Ігор Олегович, к.т.н., ст.викл., затверджені наказом по університету від «29» травня 2024р. №2178-с

2. Термін подання студентом роботи 14 червня 2024 р.

3. Вихідні дані до роботи: Різницево-далекомірна система, SDR, Raspberry Pi, Linux, Matlab.
4. Зміст роботи: 1. Актуальність, аналіз та порівняння систем визначення місцеположення джерел радіовипромінювання, 2. Аналіз та вибір апаратного забезпечення для побудови різницево-далекомірної системи, 3. Підготовка апаратних засобів та розробка програмного забезпечення для побудови різницево-далекомірної системи, 4. Експериментальне дослідження розробленого програмного забезпечення.
5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): Мультиплікативна презентація.

6. Консультанти розділів роботи*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 15.04.2024

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Аналіз технічного завдання	15.04 – 21.04	
2	Аналіз актуальності визначення місцеположення джерел випромінювання	21.04 – 28.04	
3	Огляд існуючих систем визначення місцеположення джерел випромінювання	28.04 – 05.05	
4	Аналіз апаратного забезпечення для реалізації бюджетної системи визначення джерел випромінювання	06.05 – 12.05	
5	Підготовка апаратних засобів та розробка програмного забезпечення для побудови різницево-далекомірної системи	13.05 – 26.05	
6	Експериментальне дослідження розробленого програмного забезпечення.	27.05 – 5.06	
7	Оформлення дипломної роботи	06.06 – 14.06	

Студент Антон ПАНАКОВ

Керівник Ігор ТОВКАЧ

* Якщо визначені консультанти. Консультантом не може бути зазначено керівника дипломної роботи.

АНОТАЦІЯ

Дипломна робота включає в себе: пояснювальну записку обсягом 86 сторінок, які містять 4 розділи, 47 ілюстрацій, 15 таблиць, 3 додатки, а також 31 бібліографічне найменування за переліком джерел.

Актуальність визначення місцеположення джерела радіовипромінювання є критично важливою в багатьох сферах, включаючи національну безпеку, цивільну авіацію, наукові дослідження та комерційні застосування, а розробка систем локалізації є актуальною задачею. Тому метою даної роботи є вибір апаратного забезпечення, а також розробка програмного забезпечення для створення різницево-далекомірної системи визначення місцеположення джерел радіовипромінювання на основі Raspberry Pi та SDR приймачів.

Обране апаратне забезпечення дозволяє побудувати систему, яка здатна одночасно отримувати необхідні радіосигнали та визначати часові затримки між ними за допомогою розробленого програмного забезпечення. Тестування апаратного та програмного забезпечення проводилося на експериментальному макеті.

Ключові слова: різницево-далекомірна система, Raspberry Pi, SDR, синхронізація, GPS, PPS, Wi-Fi, Matlab, Linux.

ANNOTATION

The diploma work includes: an explanatory note of 86 pages containing 4 chapters, 47 illustrations, 15 tables, 3 appendices and 31 references.

The relevance of determining the location of a radio emission source is critical in many fields, including national security, civil aviation, scientific research and commercial applications, and the development of localization systems is an urgent task. Therefore, the purpose of this work is the selection of hardware, as well as the development of software for the creation of a differential remote sensing system for determining the location of radio radiation sources based on Raspberry Pi and SDR receivers.

The selected hardware allows you to build a system that is able to simultaneously receive the necessary radio signals and determine the time delays between them using the developed software. Hardware and software testing was carried out on an experimental model.

Keywords: difference-distance measurement system, signal, Raspberry Pi, SDR, synchronization, GPS, PPS, Wi-Fi, Matlab.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

GPIO	General-Purpose Input/Output
GPS	Global Positioning System
IP	Internet Protocol
PPS	Pulse per second
RSA	Rivest, Shamir, Adleman
SD	Secure Digital
SDR	Software-defined radio
SSH	Secure Shell Protocol
TDOA	Time difference of arrival
UART	Universal asynchronous receiver/transmitter
UDEV	User/dev
SFTP	Secure File Transfer Protocol
URL	Uniform Resource Locator
USB	Universal Serial Bus
VNC	Virtual Network Computing
Wi-Fi	Wireless Fidelity
АЦП	Аналого-цифровий перетворювач
БПЛА	Безпілотний літальний апарат
ОС	Операційна система

ВСТУП

З розвитком та поширенням використання передавачів радіосигналів, а також використання безпілотних літальних апаратів, що керуються з землі, з’явилася потреба у використанні систем визначення джерел випромінювання. Необхідність створення таких систем зросла через ряд загроз, серед яких: небезпека порушення повітряного простору безпілотними літальними апаратами, незаконна передача інформації, заняття ефіру на службових або приватних частотах, порушення роботи спеціальних рятувальних служб та інші загрози, що створюються за рахунок випромінення радіохвиль.

Визначення місцеположення джерел випромінювання відіграє важливу роль у сучасному світі, переважно через незаконне використання безпілотних літальних апаратів та радіопередавачів. Через воєнні дії на території країни, актуальність визначення місцеположення джерел випромінення зросла, оскільки за допомогою різних методів радіолокації стає можливим визначення координат техніки та обладнання противника. У сучасних системах активно використовується різницево-далекомірний метод, що є різновидом пасивної локації та має перевагу над іншими системами завдяки точному визначенню місцеположення джерел випромінювання без власного випромінення сигналів, що дозволяє отримувати інформацію про положення об’єктів супротивника залишаючись непомітним для ворожих приймачів.

Існуючі системи визначення місцеположення джерел випромінювання коштують дорого, але потреба у визначенні координат джерел випромінювання залишається.

Тому метою цієї дипломної роботи є аналіз апаратного та розроблення програмного забезпечення для реалізації різницево-далекомірної системи з бюджетних компонентів.

1 Актуальність, аналіз та порівняння систем визначення місцеположення джерел радіовипромінювання

Визначення положення об’єктів завжди було актуальною темою, а з розвитком технологій визначити положення об’єктів стало можливо з високою точністю за короткий проміжок часу.

Для визначення положення уже існує безліч систем з різним цільовим призначенням, такі як, наприклад, GPS для глобального позиціонування об’єктів на поверхні Землі; Bluetooth RSSI та iBeacon для позиціонування об’єктів всередині приміщення, а також безліч радіолокаційних систем з пасивною та активною локацією для визначення положення об’єктів.

Основною загрозою є передавачі інформації, радіолокатори та безпілотні літальні апарати (БПЛА). Усі вони несуть небезпеку, але саме безпілотні літальні апарати несуть пряму загрозу, на відміну від різноманітних передавачів.

Випромінення радіохвиль є одним з найпоширеніших методів передачі інформації. Для передачі сигналів використовуються передавачі, що генерують сигнали з певною частотою та модуляцією, перетворюючи інформацію у форму, яка б могла бути передана каналом зв’язку. Оскільки канали зв’язку займають певний частотний діапазон, а кількість передавачів з кожним днем збільшується, то це призводить до обмежень доступного частотного спектру. Передавачі можуть використовувати приватні частоти або частоти, які виділені для спеціальних служб. Це призводить до того, що можуть бути створені перешкоди, які будуть погіршувати зв’язок та призводити до помилок передачі інформації. Також різноманітні приймачі можуть використовуватися для дистанційного керування пристроями та для передачі даних у незаконних цілях. Тому визначення місцеположення джерел випромінювання, яким керує порушник є важливою та актуальною темою.

1.1 Аналіз загроз використання безпілотних літальних апаратів

Розвиток сфери безпілотних літальних апаратів породжує задачу у визначенні місцеположення таких літальних апаратів. Безпілотні літальні апарати здійснюють політ та посадку без фізичної присутності пілота та можуть бути оснащені різноманітними модулями. На початку розвитку БПЛА їх використання звужувалося до військових задач, а ціна була високою для цивільного використання. Завдяки розвитку сфери мікроконтролерів та датчиків, ціни на БПЛА знизилися, а БПЛА стали більш доступними. Останні роки вони широко використовувалися в комерційних цілях, таких як фотографія, кінематограф тощо. Доступність та широкі можливості використання дозволяють застосовувати БПЛА у багатьох сферах: вирішення надзвичайних ситуацій, пошуково-рятувальні операції, охоронна та спостереження, переміщення багажу, сільськогосподарська сфера, військова сфера тощо.

Рисунок 1 – Безпілотні літальні апарати

Таке широке застосування БПЛА може спричинити ряд загроз[1]:

– Загроза для повітряного простору. Використання БПЛА біля аеродромів може спричинити зіткнення з літаками при польоті або посадці.

– Загроза падіння. Оскільки БПЛА можуть бути різних розмірів та ваги, деякі з них можуть нанести значної шкоди людині або майну.

– Загроза використання службових або приватних радіочастот. Використання БПЛА аматорами несе загрозу використання смуг частот, що використовуються спеціальним службами. Такі частоти використовуються

військовими та службами спеціального призначення. Використання цих частот є незаконним.

– Загроза вторгнення в приватне життя. Більшість БПЛА оснащені камерами, що можна використовувати для шпигунства, оскільки невеликі безпілотники непомітні та малошумні. Також вони можуть бути оснащені мікрофонами, приймачами для перехвату радіосигналів та іншими модулями, що дозволяють порушувати границі приватного життя.

– Загроза терактів. БПЛА також можуть використовуватися для здійснення терористичних актів. Їх функцією може бути як збір інформації, так і перенесення або скидання небезпечних речовин.

1.2 Аналіз методів визначення місцеположення об’єктів

Зараз ведуться активні бойові дії на території України, тому використання передавачів та систем для визначення положення об’єктів набуло більшого поширення та небезпеки.

Активно використовуються радіолокаційні станції (РЛС), що можуть визначити положення техніки та переважно літальних апаратів. Якщо виявити пасивний радіолокатор досить складно, то при активній радіолокації можливо визначити місцеположення апаратури, що випромінює сигнал.

З розвитком різноманітних передавачів та безпілотних літальних апаратів, тема визначення положення джерел випромінювання стала ще більш актуальною через їх потенційну небезпеку. Зараз визначення положення БПЛА та РЛС противника відіграє ключову роль у ефективній боротьбі з ворогом. Існують різні методи визначення положення ворожих об’єктів, що активно використовуються на даний момент.

Основними методами визначення місцеположення об’єктів є радарне виявлення та виявлення радіосигналів, які використовуються для керування та передачі даних.

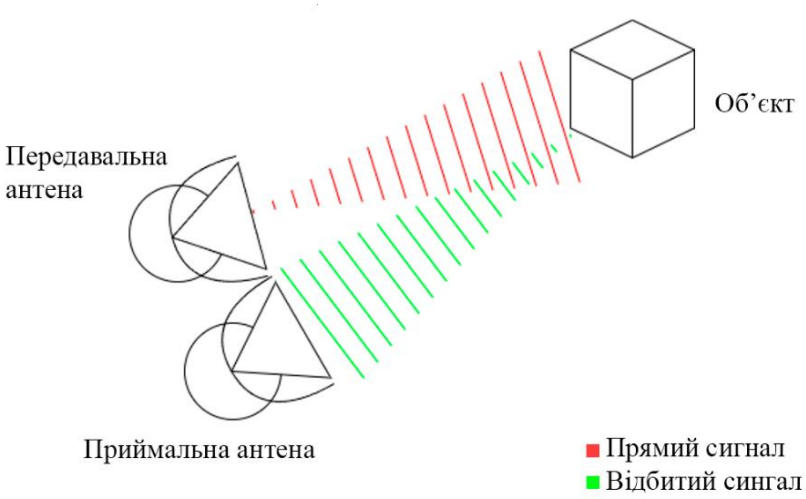

Рисунок 2 – Схематичне пояснення роботи активної локації

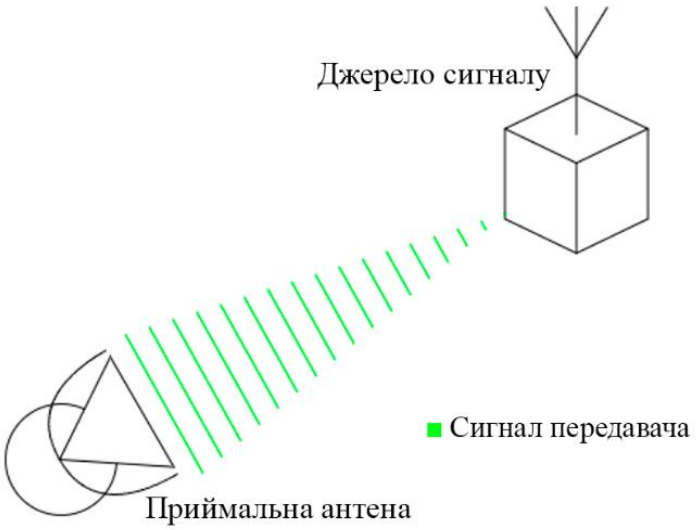

Рисунок 3 – Схематичне пояснення роботи пасивної локації

Активна радіолокація досить небезпечна, оскільки противник може визначити джерело сигналу і відповідно місцезнаходження радіолокатора. Пасивна локація відноситься до методів визначення місцезнаходження об’єкта без використання активних радіосигналів або іншого видимого випромінювання. Замість цього, пасивні локатори використовуються для реєстрації та аналізу різних типів електромагнітних сигналів, таких як радіохвилі, оптичне випромінювання, акустичні хвилі тощо. Пасивна локація може принести значні переваги з точки зору безпеки та протидії загрозам.

1.3 Аналіз методів пасивної локації

Серед поширених методів пасивної локації існує кутомірний, різницево-далекомірний та кутомірно-різницево-далекомірний метод. Кутомірний метод оснований на визначенні кута випромінювання сигналу до кількох приймачів та дозволяє визначити напрямок на джерело, але цей метод не здатен визначати відстань до джерела випромінювання. За допомогою різницево-далекомірної системи можливо визначити кутові координати та дальність до об’єкта випромінювання, що дає можливість точного визначити місцеположення джерела випромінювання. Кутомірно-різницево-далекомірний метод поєднує в собі два попередні методи, але досить складний в реалізації. На основі цього, найоптимальнішим методом визначення місцеположення джерела випромінювання є різницево-далекомірний метод.

Різницево-далекомірний метод визначає місцеположення джерел випромінювання за допомогою визначення різниці в часі проходження сигналу до кількох приймачів. Для реалізації системи потрібно мінімум 3 приймачі. Порядок визначання місцеположення джерела випромінювання полягає в прийомі кількома приймачами сигналу, вимірювання різниці в часі надходження сигналів до цих приймачів, за різницею часу поширення сигналу будуються гіперболічні лінії, а перетин цих гіперболічних ліній дозволяє визначити місцеположення джерела випромінювання. Такий метод має певні переваги, серед яких є висока точність та можливість пасивного виявлення без вимірювання відстані. Головними недоліками систем, що використовують різницево-далекомірний метод, є необхідність використання мінімум трьох рознесених приймачів та вимога високої точності синхронізації приймачів.

Оскільки потреба визначення місцеположення джерел випромінювання з’явилася ще на початку розвитку передавачів, то уже існують комплексні системи визначення положення джерел випромінювання.

1.4 Аналіз методів визначення місцеположення джерел випромінювання

Поширення передавачів та БПЛА несе загрозу, тому постає задача визначення місцезнаходження таких пристроїв. Якщо визначити місцеположення літальних апаратів є більш складною задачею, ніж визначення місцеположення наземних джерел випромінювання, то основна увага буде приділена визначенню координат літальних апаратів, а саме БПЛА.

Найкраще використовувати радіолокаційні системи з пасивною локацією для виконання задачі визначення положення джерел випромінювання. Оскільки активні радіолокатори приймають відбитий сигнал від об’єктів та отримують інформацію про позиціонування цих об’єктів, а отримати відбитий сигнал від, наприклад, БПЛА досить складна задача через невеликі габарити літальних апаратів, то доцільним буде використання саме методів пасивної локації. Зазвичай БПЛА керується з землі та передає інформацію на пульт керування. Виявити сигнали, що передає БПЛА значно легше та доцільніше, ніж виявити сам БПЛА. Тому задача визначення місцезнаходження БПЛА за сингалами, що він випромінює, найкраще буде реалізована методом пасивної радіолокації.

Визначення положення радіолокаційної станції також непроста задача, але за аналогією, можна виявити сигнал активного локатора та визначити місцезнаходження обладнання.

Необхідність розробки систем виявлення місцеположення безпілотних літальних апаратів призвело до появи комплексних систем для визначення параметрів БПЛА та їх місцезнаходження.

Класифікують системи виявлення БПЛА на такі види[2]:

– **Акустичні (звукові)** – контролюють спектри звукових частот, характерні для БПЛА. Вони дозволяють виявити БПЛА в безпосередній близькості на відстані кілька сотень метрів. Але детектор звуку не працює в міських умовах.

– **Радіочастотні** – аналізують радіохвильові сигнали в діапазонах частот, на яких відбувається керування БПЛА (включає в себе моніторинг частот). Такі пристрої мають більшу дальність виявлення.

– **Візуальні (оптичні)** – виявляють БПЛА за допомогою камер спостереження, які знаходять рухомий повітряний об'єкт, намагаючись диференціювати БПЛА і птахів на основі розміру, траєкторії польоту і стилю руху. Камери можуть побачити на відстані не більше 350 метрів. Навіть використовуючи комп'ютерні алгоритми, які відстежують політ моделі, визначити птах це або БПЛА буде дуже складно.

– **Теплові** – визначають сигнатуру тепла БПЛА, маючи ефективний діапазон близько 350 метрів для виявлення, але тепловізори менш ефективні, ніж звукові датчики. При цьому більшість БПЛА виготовлено зараз із пластику з електродвигунами, що випромінюють мало тепла, навіть менше ніж птахи, що робить цей метод абсолютно неефективним.

– **Радарні** – більш досконалі пристрої виявлення БПЛА використовують технології ближньої радіолокації. Використані технології ближньої радіолокації дають максимально повну інформацію про рух об'єкта, що відстежується: місце виявлення, траєкторію руху місцевістю або акваторією, швидкість руху, розміри. Фіксуються кадри відеоспостереження, в тому випадку, якщо є додаткова поворотна платформа з відеокамерою або тепловізором.

1.5 Аналіз існуючих систем визначення положення джерел випромінювання

Drone Detection System від Dedrone – це комплексне рішення для виявлення, ідентифікації та протидії безпілотним літальним апаратам. Здатна визначати місцезнаходження, висоту, швидкість та траєкторію польоту БПЛА також класифікує тип безпілотника та ідентифікує оператора. Має наступні характеристики, наведені в таблиці 1.1[3].

Таблиця 1.1 – Характеристика Drone Detection System

Дальність дії	До 5 км для радіочастотних сенсорів, до 2 км для оптичних та акустичних сенсорів
Спосіб виявлення	Поєднання радіочастотного, оптичного та акустичного моніторингу
Ціна	Від $50,000 до $150,000 в залежності від конфігурації та кількості сенсорів

Рисунок 4 – Drone Detection System

DroneShield DroneSentinel – це інтегрована система виявлення та протидії безпілотним літальним апаратам. Визначає координати, висоту, швидкість та напрямок руху БПЛА. Характеристика наведена в таблиці 1.2[4].

Таблиця 1.2 – Характеристика DroneShield DroneSentinel

Дальність дії	До 5 км для радіочастотних, акустичних та оптичних сенсорів
Спосіб виявлення	Поєднання радіочастотного, акустичного та оптичного моніторингу
Ціна	Від $70,000 до $150,000 залежно від кількості сенсорів та додаткових функцій

Рисунок 5 – DroneShield DroneSentinel

DEDRONE DRONETRACKER – це інтегрована система виявлення та протидії безпілотним літальним апаратам. Характеристика наведена в таблиці 1.3[5].

Таблиця 1.3 – Характеристика DroneTracker

Дальність дії	До 500 м
Спосіб виявлення	Поєднання радіочастотного, акустичного, оптичного моніторингу та інфрачервоної детекції (ближня область ІЧ-спектру, від 0,75 до 3 мкм). Також: Виявлення Wi-Fi (від 2,4 ГГц до 5 ГГц ISM)
Кут охоплення оптичної камери	10° - 90°
Ціна	Від $50,000

Рисунок 6 – Система DroneTracker

Система CerbAir – це інтегрований комплекс для виявлення, ідентифікації та протидії безпілотним літальним апаратам. Здатність класифікувати понад 1500 моделей безпілотників, а час реакції виявлення менше 2 секунд. Характеристика наведена в таблиці 1.4[6].

Таблиця 1.4 – Характеристика CerbAir

Дальність дії	До 3 км для радіочастотних сенсорів, до 150 м для оптичних та до 100 м для тепловізійних сенсорів
Спосіб виявлення	Поєднання радіочастотного, акустичного, оптичного та тепловізійного моніторингу
Ціна	Від €200,000

Рисунок 7 – Система CerbAir

Усі ці системи унікальні та мають певні особливості у вигляді різних датчиків, дальності дії та інших параметрів систем. Головні недоліки розглянутих систем – це ціна на такі системи та закритий програмний код.

1.6 Висновок до розділу 1

Джерела випромінювання несуть різноманітні загрози, тому постає задача визначення положення цих джерел випромінювання. Виконують цю задачу системи визначення положення джерел випромінювання, але вони дорогі. Тому був проведений аналіз компонентів для побудови бюджетної системи, яка могла б могла визначати місцеположення джерел сигналу.

2 Аналіз та вибір апаратного забезпечення для побудови різницево-далекомірної системи

Для побудови системи визначення місцеположення джерел сигналу потрібно, перш за все, визначитися з методом отримання сигналу. Аналіз показав, що найкраще цю задачу виконує радіочастотний моніторинг, оскільки радіочастотний діапазон охоплює широкий спектр від кГц до ГГц, що дозволяє виявляти різноманітні типи сигналів. При радіочастотному моніторингу за допомогою сучасних радіочастотних приймачів, можливо отримати високу чутливість, що дозволяє виявляти навіть слабкі сигнали, зо знаходяться на великій відстані або замасковані перешкодами. Серед переваг радіочастотного методу також можна виділити здатність обробки радіочастотних сигналів з високою швидкістю, що дозволяє здійснювати оперативне виявлення та локалізацію джерел сигналів. Здатність здійснювати пасивну локацію, тобто визначати місцезнаходження джерела сигналу без його активного опромінення забезпечує радіочастотний метод виявлення. Така пасивність важлива в ситуаціях, коли активний вплив на об'єкт небажаний або неможливий.

Отже, зупинимося на радіоприймачах, які будуть визначати параметри системи виявлення місцеположення джерел сигналів. Радіоприймач – це пристрій, призначений для приймання електромагнітних хвиль радіодіапазону з наступним перетворенням інформації, яка у них зберігається до вигляду, в якому вона може бути використана[7].

Найкраще використовувати Software Defined Radio (SDR) приймачі. Software Defined Radio – це система радіозв'язку, в якій програмне забезпечення використовується як для модуляції, так і для демодуляції радіосигналів. При використанні SDR персональний комп'ютер стає ядром любительської радіостанції, завдяки чому практично весь обсяг робіт із обробки сигналу перекладається на програмне забезпечення, яке запускається на персональному комп'ютері[8].

Мета такого підходу - створити систему, яка може приймати і передавати практично будь-які радіосигнали за допомогою програмного забезпечення, що є гнучким і адаптивним.

Перевагою таких приймачів є те, що основні радіоприймальні функції реалізовані за допомогою програмного забезпечення, а не жорсткої апаратної логіки, що дозволяє легко перепрограмовувати їх для роботи з різними видами модуляції, протоколами зв'язку та діапазонами частот без зміни апаратної частини. SDR приймачі здатні охопити значно ширший діапазон частот (від десятків кГц до кількох ГГц) в порівнянні з традиційними приймачами, а це робить їх придатними для роботи з широким спектром радіосигналів. SDR приймачі здатні виконувати одночасно функції радіоприймача, аналізатора спектру, моніторингу зв'язку та інші завдання, що забезпечує більшу універсальність та ефективність використання.

Найголовнішою перевагою SDR приймачів для реалізації на їх основі системи визначення положення джерел випромінювання є їх відкритий вихідний код, що дає змогу користувачам адаптувати приймачі під свої потреби.

Серед популярних SDR радіоприймачів, які можна придбати, є декілька варіантів, досить схожих між собою за технічними характеристиками та ціною.

2.1 Аналіз бюджетних приймачів сигналів, які можливо використати при побудові системи виявлення джерел випромінювання

HackRF One – популярний приймач, створений для розробки та тестування радіотехнічних проектів. HackRF One має відкриту апаратну платформу, яка використовується як USB-периферія. Пристрій може приймати радіосигнали в діапазоні від 1МГц до 6ГГц, а також передавати їх з пропускною здатністю до 20МГц. Характеристики наведені в таблиці 2.1[9].

Таблиця 2.1 – Характеристика HackRF One

Робоча частота	Від 1 МГц до 6 ГГц
Частота дискретизації	8-бітна квадратура
Смуга пропускання	20 МГц
Ціна	4985 грн

Рисунок 8 – HackRF One SDR

Airspy R2 – це удосконалений, з відкритим вихідним кодом, програмний радіоприймач, який має прийнятний динамічний діапазон. Він здатний виконувати вибірку спектра в діапазоні від 10 МГц, в межах частот від 24 МГц до 1,8 ГГц, а також за їх межами. Характеристики наведені в таблиці 2.2[10].

Таблиця 2.2 – Характеристика Airspy R2

Робоча частота	Від 24 МГц до 1.8 ГГц
Частота дискретизації	12 біт
Смуга пропускання	10 МГц
Ціна	11180 грн

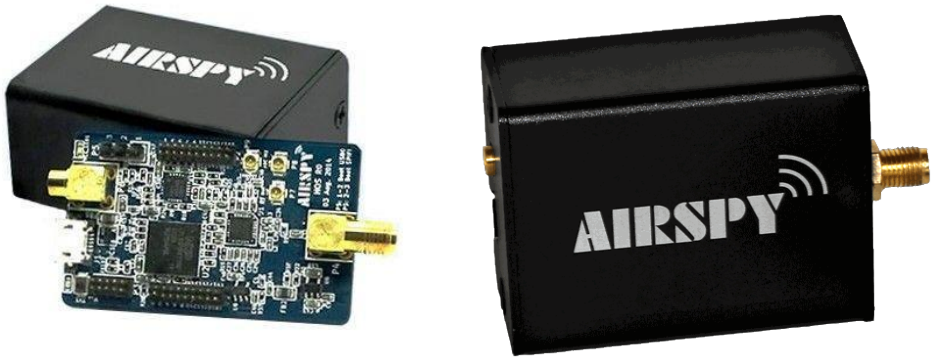

Рисунок 9 – Airspy R2

ADALM-PLUTO – це платформа, розроблена Analog Devices, призначена для активного навчання у сфері радіоелектроніки та бездротового зв'язку. Ключовим компонентом є апаратний модуль PlutoSDR – високопродуктивний радіочастотний пристрій, який може працювати в різних частотних діапазонах. PlutoSDR дозволяє користувачам вивчати, експериментувати та вдосконалювати свої знання у цій галузі. Ця платформа призначена для використання в лабораторних, дослідницьких та експериментальних цілях, включаючи аматорські радіотехнічні проекти, наприклад, для передачі на каналі висхідного потоку 2,4 ГГц. Характеристики наведені в таблиці 2.3[11].

Таблиця 2.3 – Характеристика ADALM-PLUTO

Робоча частота	від 325 МГц до 3,8 ГГц
Частота дискретизації	12 біт
Смуга пропускання	2,4 МГц
Ціна	18000 грн

Рисунок 10 – ADALM-PLUTO

LimeSDR – це відкрита та недорога платформа програмно-визначуваного радіозв'язку (SDR), яка забезпечує широку підтримку різноманітних бездротових стандартів та протоколів. Ця гнучка платформа дозволяє розробникам та ентузіастам експериментувати та створювати додатки. Завдяки відкритому вихідному коду та всебічній підтримці різних стандартів, LimeSDR є потужним інструментом для дослідження, розробки та використання широкого спектру бездротових рішень. Характеристики наведені в таблиці 2.4[12].

Таблиця 2.4 – Характеристика LimeSDR

Робоча частота	100 кГц – 3,8 ГГц
Частота дискретизації	12 біт
Смуга пропускання	61,44 МГц
Ціна	30000 грн

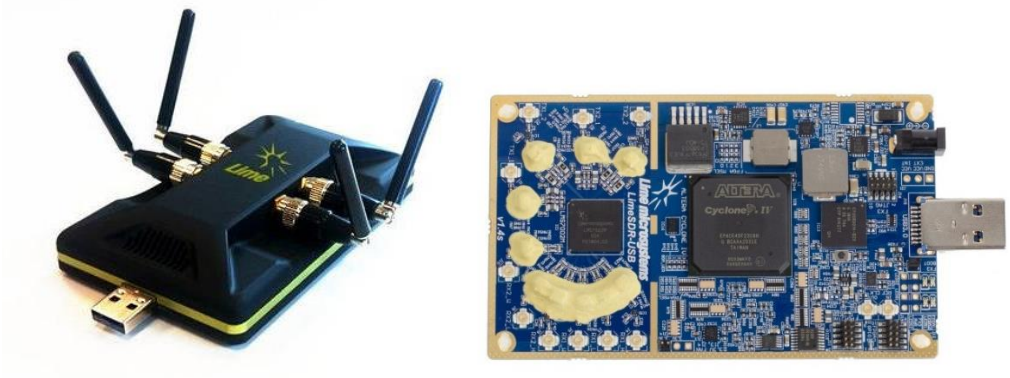

Рисунок 11 – LimeSDR

Наведені радіоприймачі задовольняють потреби при побудові системи визначення джерел випромінювання, навіть мають зайві функції, наприклад не тільки прийом, а й передача сигналів, що збільшує вартість таких приймачів.

При аналізі ринку радіоприймачів, найоптимальнішим SDR радіоприймачем обрано RTL-SDR V3:

RTL-SDR V3 – це широкосмуговий програмно-визначуваний радіоприймач, що підтримує аналогові та цифрові сигнали. Він сумісний з Windows, Linux, Android та Mac OS X, що робить його універсальним інструментом для широкого спектру радіочастотних проектів. Характеристики наведені в таблиці 2.5[13]:

Таблиця 2.5 – Характеристика RTL-SDR V3

Робоча частота	500 кГц – 1,766 ГГц
Частота дискретизації	8 біт
Смуга пропускання	До 2,4 МГц
Ціна	1500 грн

Рисунок 12 – RTL-SDR V3

Це найкращий варіант серед представлених. Серед переваг можна виділити наступне:

– доступна ціна;

– широкий діапазон частот;

– підтримка багатьох ОС, що забезпечує гнучкість використання;

– активна спільнота розробників.

Звичайно, за невеликі кошти, отримати ідеальний SDR радіоприймач не можливо, тому є і недоліки:

– Обмежена чутливість порівняно з дорогими SDR приймачами, RTL-SDR v3 має нижчу чутливість та динамічний діапазон.

– Менша роздільна здатність АЦП - RTL-SDR v3 має 8-бітний АЦП, в той час як професійні SDR можуть мати 12-бітні або 16-бітні АЦП для кращої динаміки.

– Відсутність вбудованих фільтрів - RTL-SDR v3 не має вбудованих смугових фільтрів, що може вимагати додаткового фільтрування сигналу

Не зважаючи на недоліки, для реалізації апаратного та програмного забезпечення різницево-далекомірної системи такі характеристики приймачів будуть достатніми, а зважаючи на те, що при побудові різницево-далекомірної системи потрібно мінімум три приймача, то це найвигідніше рішення. У порівнянні з професійними SDR приймачами, ціна системи буде відрізнятися приблизно в 10 разів.

2.2 Аналіз мікрокомп’ютерів для керування та живлення приймача RTL-SDR v3

Зазвичай SDR приймачі підключаються до персонального комп’ютера для керування, але оскільки задача стоїть в побудові різницево далекомірної системи, яка передбачає віддалене розміщення приймачів один від одного для широкого охоплення площі визначення місцеположення джерел

випромінювання, то використання громіздких персональних комп’ютерів буде недоцільним.

Використання персональних комп’ютерів для керування та живлення має ряд недоліків, серед яких:

– Великі габарити, які ускладнюють переміщення обладнання, що не вписується в рамки мобільної системи визначення положення джерел сигналів.

– Велике споживання. Персональні комп’ютери потребують постійного підключення до мережі. Навіть при використанні ноутбуків, автономність системи буде досить малою.

– Ціна на персональні комп’ютери. Для реалізації системи визначення джерел випромінювання різницево-далекомірним методом потребує мінімум 3 персональні комп’ютери, що досить дороговартісно, з урахуванням того, що однією з цілей роботи є реалізація повноцінної системи набагато дешевше аналогів.

Шляхом вирішення наявної проблеми може слугувати використання мікрокомп’ютерів. Необхідно проаналізувати наявні мікрокомп’ютери, що б задовольняли потреби системи визначення положення джерел випромінювання з використанням RTL-SDR v3.

Мікрокомп'ютери – це компактні обчислювальні пристрої, реалізовані на одній системі-на-кристалі (System-on-a-Chip). Ці пристрої характеризуються невеликими фізичними розмірами, низьким енергоспоживанням та продуктивністю, порівнянною із сучасними планшетами і смартфонами. Однак, на відміну від портативних пристроїв, мікрокомп'ютери зазвичай не мають вбудованих екранів, але оснащені HDMI-відеовиходом та відкритими операційними системами. Вони часто використовуються як навчальні комп'ютери, домашні сервери або вбудовані контролери в різноманітних проектах[14].

Найперше, що приходить на думку при згадуванні мікрокомп’ютерів – це мікрокомп’ютери Raspberry Pi.

Raspberry Pi – це популярна лінійка одноплатних мікрокомп'ютерів, вперше представлена у 2013 році. Початкова модель Raspberry Pi була побудована на системі-на-чипі (SoC) Broadcom BCM2835, яка включала процесор ARM із тактовою частотою 700 МГц, графічний процесор VideoCore IV та 512/256 Мб оперативної пам'яті. Замість жорсткого диску, ці пристрої використовують мікро-SD картки для зберігання даних та операційних систем. Незважаючи на скромні апаратні характеристики, Raspberry Pi забезпечував можливість відтворення відео у роздільній здатності 1080p. Для цих мікрокомп'ютерів був розроблений спеціалізований дистрибутив Linux, відомий як Raspbian OS, заснований на Debian [15].

Характеристики наведені в таблиці 2.6 для Raspberry Pi 3 Model B+ та в таблиці 2.7 для Raspberry Pi 4 Model B.

Raspberry Pi 3 Model B+ – це компактний одноплатний комп'ютер, що має розміри порівняні з банківською картою. Ця модель є удосконаленою версією попереднього Raspberry Pi 3 Model B. Raspberry Pi 3 Model B+ базується на системі-на-чіпі BCM2837B0, яка включає 4-ядерний 64-бітний ARM-процесор з тактовою частотою 1,4 ГГц, а також потужний відеопроцесор VideoCore IV.

Рисунок 13 – Raspberry Pi 3 Model B+

Таблиця 2.6 – Характеристика Raspberry Pi 3 Модель B+

Процесор (CPU)	64-бітний 4-ядерний ARMv8 Cortex-A53 із тактовою частотою 1,4 ГГц
Відеопроцесор (GPU)	VideoCore IV 3D
Оперативна пам’ять (RAM)	1 Гб LPDDR2 (900 МГц)
Периферія	– аудіо/відео цифровий: HDMI – аудіо/відео аналоговий: 3,5 мм jack (4 pin) – USB порт: USB 2.0 - 4 шт – накопичувач: microSD – мережа: 10/100 Ethernet – порт дисплея: Display Serial Interface (DSI) – порт камери: MIPI Camera Serial Interface (CSI-2) – введення-виведення: GPIO 40 pin
Бездротові інтерфейси	Wi-Fi: двосмуговий 2,4 ГГц і 5 ГГц, Bluetooth: 4.2 Classic і Low Energy (BLE)
Ціна	$50

Raspberry Pi 4 Model B – це нова та покращена версія в популярній лінійці одноплатних мікрокомп'ютерів Raspberry Pi, що має ряд суттєвих удосконалень у порівнянні з попередніми поколіннями: значне підвищення швидкості та продуктивності процесора, що базується на більш потужному ARM-ядрі; покращення продуктивності мультимедійного модуля, що дозволяє відтворювати відео у ще вищій якості; збільшення обсягу та швидкості оперативної пам'яті, що забезпечує кращу продуктивність при виконанні ресурсоємних задач; оновлені мережеві модулі, що пропонують сучасні стандарти підключення. Попри ці суттєві апаратні модернізації, Raspberry Pi 4 Model B зберігає звичний рівень енергоспоживання та сумісність з більшістю периферійних пристроїв, використовуваних для попередніх версій, що дозволяє оновлювати існуючі проекти на базі Raspberry Pi, отримуючи при цьому відчутний приріст продуктивності. Загалом цей мікрокомп’ютер є найбільш поширеним та популярним для використання в будь-яких проектах, аноді використовується як основний комп’ютер для роботи деякими ентузіастами. Характеристики наведені в таблиці 2.8.

Рисунок 14 – Raspberry Pi 4 Model B

Таблиця 2.7 – Характеристика Raspberry Pi 4 Model B

Процесор (CPU)	Broadcom BCM2711, чотирьохядерний процесор Cortex-A72 (ARM v8), 64-розрядний процесор на частоті 1,5 ГГц
Відеопроцесор (GPU)	OpenGL ES 3.0
Оперативна пам’ять (RAM)	8GB LPDDR4-2400 SDRAM
Периферія	– Гігабіт Ethernet – 2 порти USB 3.0 – 2 порти USB 2.0. – 2 × порти micro-HDMI (підтримується до 4kp60) – 2-канальний MIPI DSI порт дисплея – 2-канальний порт камери MIPI CSI – 4-полюсний стереоаудіо та композитний відеопорт – введення-виведення: GPIO 40 pin
Бездротові інтерфейси	2,4 ГГц і 5,0 ГГц IEEE 802.11ac бездротовий, Bluetooth 5.0, BLE
Ціна	$100

Ще однією популярною лінійкою мікрокомп’ютерів є лінійка Orange Pi. Розглянемо найпопулярніший мікрокомп’ютер Orange Pi 5 8G.

Orange Pi 5 – це одноплатний комп'ютер, що працює під керуванням операційної системи Linux. Він дуже компактний та має розміри 62X100 мм.

Orange Pi 5 оснащений потужним 8-ядерним 64-бітним процесором Rockchip RK3588S, що працює на частоті 2,4 ГГц. Він має 8 ГБ оперативної пам'яті LPDDR4 SDRAM та 16 ГБ вбудованої флеш-пам'яті eMMC. Плата

пропонує широкий набір портів та інтерфейсів, зокрема HDMI-вихід, 26-контактний GPIO-роз'єм, гігабітний Ethernet, 1 порт USB 3.0, 2 порти USB 2.0, слот M.2 M-KEY та MIPI CSI для підключення камери. Для живлення Orange Pi 5 можна використовувати будь-який блок живлення на 5 В, 4 А з роз'ємом Type-C. Ця одноплатна обчислювальна система добре підходить для мультимедійних та інших ресурсномістких застосунків. Характеристики наведені в таблиці 2.8[16].

Рисунок 15 – Orange Pi 5

Таблиця 2.8 – Характеристика Orange Pi 5

Процесор (CPU)	Rockchip RK3588S 8-core 64-bit (8nm LP процес)
Відеопроцесор (GPU)	Mali-G610 MP4 Сумісний з OpenGL ES1.1/2.0/3.2, OpenCL 2.2 та Vulkan 1.2
Оперативна пам’ять (RAM)	8GB (LPDDR4/4x)
Периферія	– HDMI2.1, до 8K при 60 Гц

	– DP1.4 (DisplayPort), DP1.4 and USB3.1 (порти мультиплексовані, і порт використовується спільно з Type-C) – 2 x MIPI D-PHY TX 4Lane, до 4K @60Hz – 3.5mm (вхід/вихід) – Кодек: ES8388 – Мікрофон – 1 x USB3.0 – 2 x USB2.0 – 1 x Type-C (USB3.1) – GPIO 26 контактний гребінець (UART, PWM, I2C, SPI, CAN)
Бездротові інтерфейси	2,4 ГГц і 5,0 ГГц IEEE 802.11ac бездротовий, Bluetooth 5.0, BLE
Ціна	$125

Інші виробники та моделі мікрокомп’ютерів мають значні недоліки у вигляді недосконалої периферії, великої ціни та найважливіше – поширеність драйверів, адаптація, універсальність та багатозадачність. Інші виробники не можуть забезпечити достатньо для реалізації різницево-далекомірної системи на основі RTL-SDR v3 без самостійного створення драйверів та адаптації модулів, що значно ускладнило б роботу з ними.

Порівнюючи Raspberry Pi 4 Model B та Orange Pi 5, то Raspberry є більш потужною та сучасною платформою для реалізації різницево-далекомірної системи на основі RTL-SDR v3. Raspberry Pi 4 Model B має потужніший процесор та кращу графічну підсистему, має широку лінійку оригінальних та недорогих модулів. Також на платформі Raspberry більше користувачів, що

вдосконалюють драйвери, адаптують периферію для мікрокомп’ютерів та залишають свої дослідження в загальному доступі.

Orange Pi 5 є непоганим аналогом для реалізації цієї роботи, але збірка та встановлення драйверів є більш складною, ніж на Raspberry Pi 4. Тому, виходячи з аналізу мікрокомп’ютерів найбільш вдалим варіантом буде використання Raspberry Pi 4 Model B.

2.3 Аналіз модулів для визначення положення мікрокомп’ютера

Оскільки ми визначилися з вибором мікрокомп’ютера, а саме Raspberry Pi 4, тому потрібно проаналізувати наявні модулі для визначення просторового положення, а саме модулі GPS. Визначення просторового положення мікрокомп’ютерів необхідне для синхронізації мікрокомп’ютерів, оскільки різницево-далекомірна система вимагає якомога точнішої синхронізації. Така скрупульозна синхронізація необхідна для точного визначення кореляції сигналів та подальшого визначення часових затримок і відповідно визначення положення джерела випромінювання за допомогою апаратного та програмного забезпечення різницево-далекомірної системи. У цьому підрозділі проведено огляд популярних GPS модулів, що сумісні для використання з мікрокомп’ютерами Raspberry Pi.

Raspberry Pi LC29H(AA) GPS HAT – це високоточний позиціонуючий модуль, спеціально розроблений для Raspberry Pi. Він використовує технологію двосмугового GPS для отримання надзвичайно точних географічних координат. Модуль також підтримує функцію RTK (Real-Time Kinematic), що робить його ідеальним вибором для застосунків, які потребують високої точності та надійності позиціонування, таких як навігація, картографування, безпілотні транспортні засоби та багато інших. Характеристика наведена в таблиці 2.9[17].

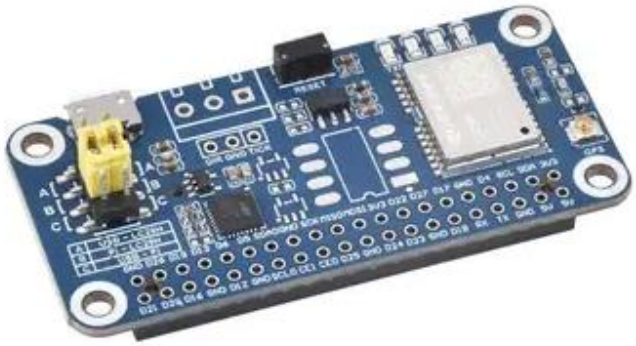

Рисунок 16 – Raspberry Pi LC29H(AA) GPS HAT

Таблиця 2.9 – Характеристика Raspberry Pi LC29H(AA) GPS HAT

Особливості	– Має стандартний роз'єм розширення Raspberry Pi 40PIN GPIO – Підтримує одночасне відстеження дводіапазонних супутникових сигналів – Підтримує одночасний прийом кількох систем GNSS (GPS, BDS, GLONASS, Galileo та QZSS) – Підтримує технологію EASY, щоб реалізувати позиціонування за допомогою збереженої інформації – Підтримує активне активне придушення перешкод, ефективно пригнічуючи або усуваючи перешкоди вузькосмугового сигналу (Wi-Fi/2/3/4/5G) для забезпечення точності навігації

Модуль GPS NEO-6M – плата на основі GPS модуля NEO-6M. GPS навігація застосовна у безлічі DIY проектів, цей модуль дозволить максимально спростити завдання зі збирання пристрою та відразу приступити до роботи. Характеристика наведена в таблиці 2.10[18].

Рисунок 17 – GPS NEO-6M

Таблиця 2.10 – Характеристика NEO-6M

Особливості	– Використовує SBAS (Satellite Based Augmentation System) – супутникові системи диференціальної корекції, що збільшують точність визначення положення до 2м. – Низьке енергоспоживання – Наявність RTC для синхронізації часу за протоколом NTP

L76X GPS HAT – це модуль, що сумісний з Raspberry Pi, Jetson Nano та багатьма іншими мікрокомп’ютерами. Він підтримує системи позиціонування GPS, BD2, PPS та QZSS. Переваги модуля – компактні розміри, низьке енергоспоживання та швидке позиціонування. Завдяки цьому модулю стає можливим розробка програми глобального позиціонування, яка підтримує алгоритм високоточної калібрування карт Baidu. Швидке визначення місцеположення, що відбувається менше ніж за 1 секунду. Цей модуль ідеально підходить для додатків, які вимагають високоточного позиціонування, таких як навігація, роботизовані системи, картографування тощо. Характеристика наведена в таблиці 2.11[19].

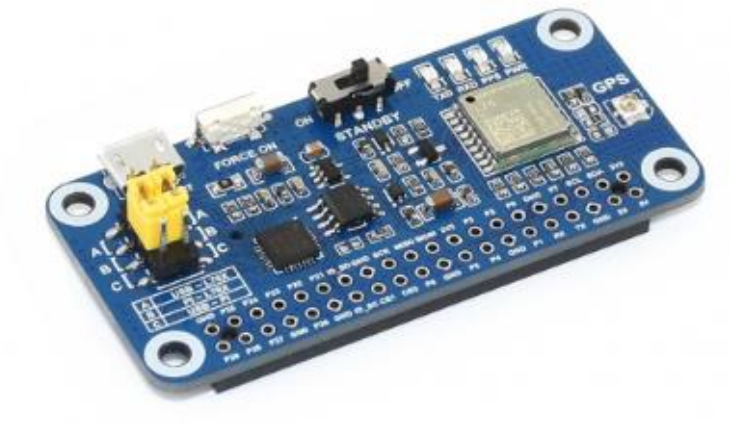

Рисунок 18 – L76X GPS HAT

Таблиця 2.11 – Характеристика L76X GPS HAT

Особливості	– Канал прийому: 33 канали відстеження, 99 каналів захоплення – Частота захоплення: -148 дБм – Частота відстеження: -163 дБм – Швидкість передачі UART: 4800~115200 (за замовчуванням 9600) – Частота оновлення: Макс. 10 Гц (за замовчуванням 1 Гц) – Робоча напруга: 2,7 В ~ 5 В (вхід живлення від контакту 5 В) – Робочий струм: 13 мА – Прийом сигналу: GPS, BD2 і QZSS – Точність позиціонування: <2,5mCEP – Максимальна швидкість: 515 м/с

Проаналізувавши популярні модулі для Raspberry Pi, можемо обрати найкращий з запропонованих. Найвищу точність визначення координат забезпечує модуль L76X GPS HAT, хоча NEO-6M має приємну ціну, проте в цьому випадку точність важливіше через потребу в точній синхронізації. Тому найкращим рішенням буде використати L76X Multi-GNSS HAT, адже окрім

високої точності до 1 метра, що досягається з використанням більш досконалого чіпу, також у цього модуля є підтримка GLONASS, BeiDou та QZSS, що дозволяє отримувати сигнали з більшої кількості супутників, покращуючи доступність та точність. Ще однією перевагою цього модуля є більш широке налаштування через додаткові можливості програмування.

2. 4 Висновок до розділу 2

Для побудови системи визначення місцеположення джерел випромінювання було обрано такі апаратні компоненти:

– Приймач RTL-SDR v3
– Мікрокомп’ютер Raspberry Pi 4
– Модуль L76X GPS HAT

Характеристики та можливості такого апаратного забезпечення дозволяють реалізувати різницево-далекомірну систему для визначення місцеположення джерел випромінювання.

3 Підготовка апаратних засобів та розробка програмного забезпечення для побудови різницево-далекомірної системи

Після визначення апаратного забезпечення, стає можливим створення макету різницево-далекомірної системи для налаштування компонентів та розробки програмного забезпечення для побудови різницево-далекомірної системи. Блок-схема системи зображена на рис. 19.

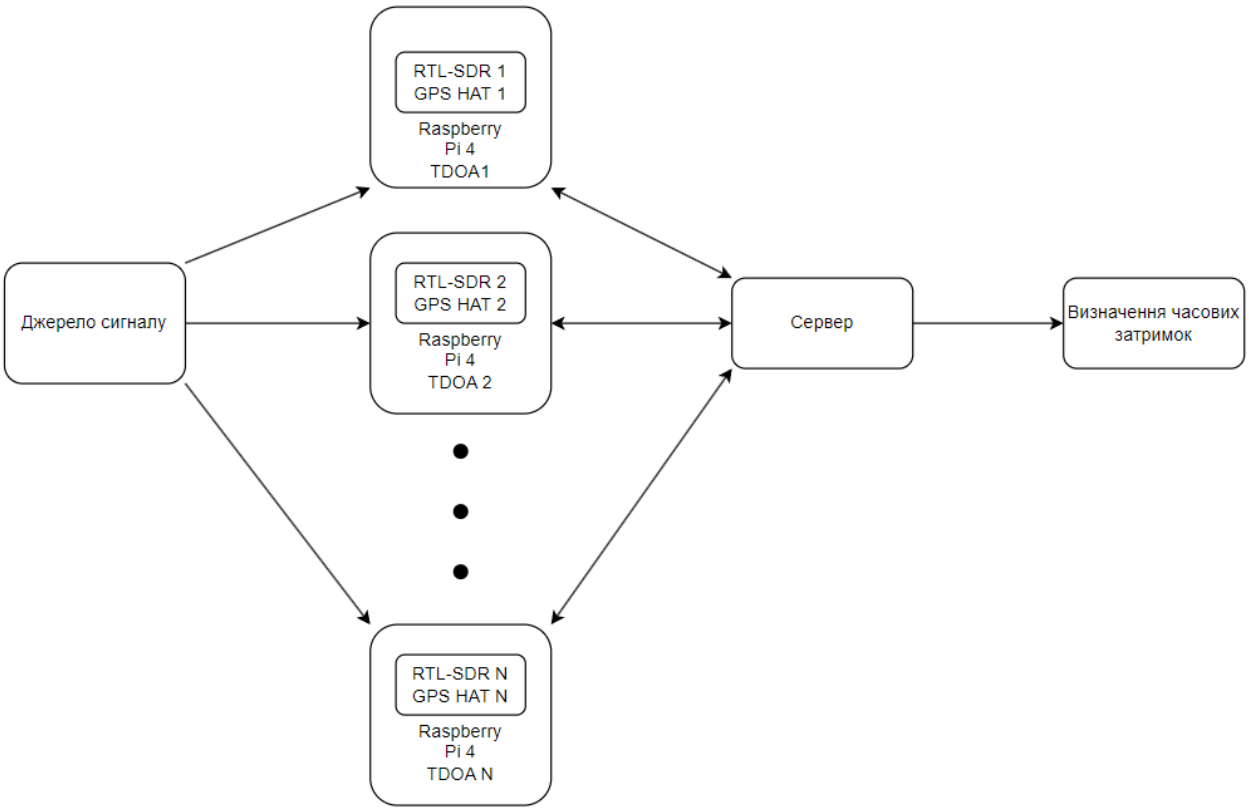

Рисунок 19 – Блок-схема різницево-далекомірної системи

За цією блок-схемою був побудований макет системи для розробки та тестування цієї системи з мінімальної кількості приймачів. Макет складається з трьох мікрокомп’ютерів Raspberry Pi, трьох приймачів RTL-SDR v3, трьох модулів L76X GPS HAT та локальної мережі. Приймачі та модулі GPS під’єднані до мікрокомп’ютерів за допомогою USB, самі ж мікрокомп’ютери з’єднані за допомогою Ethernet. Зображення макету показано на рис. 20.

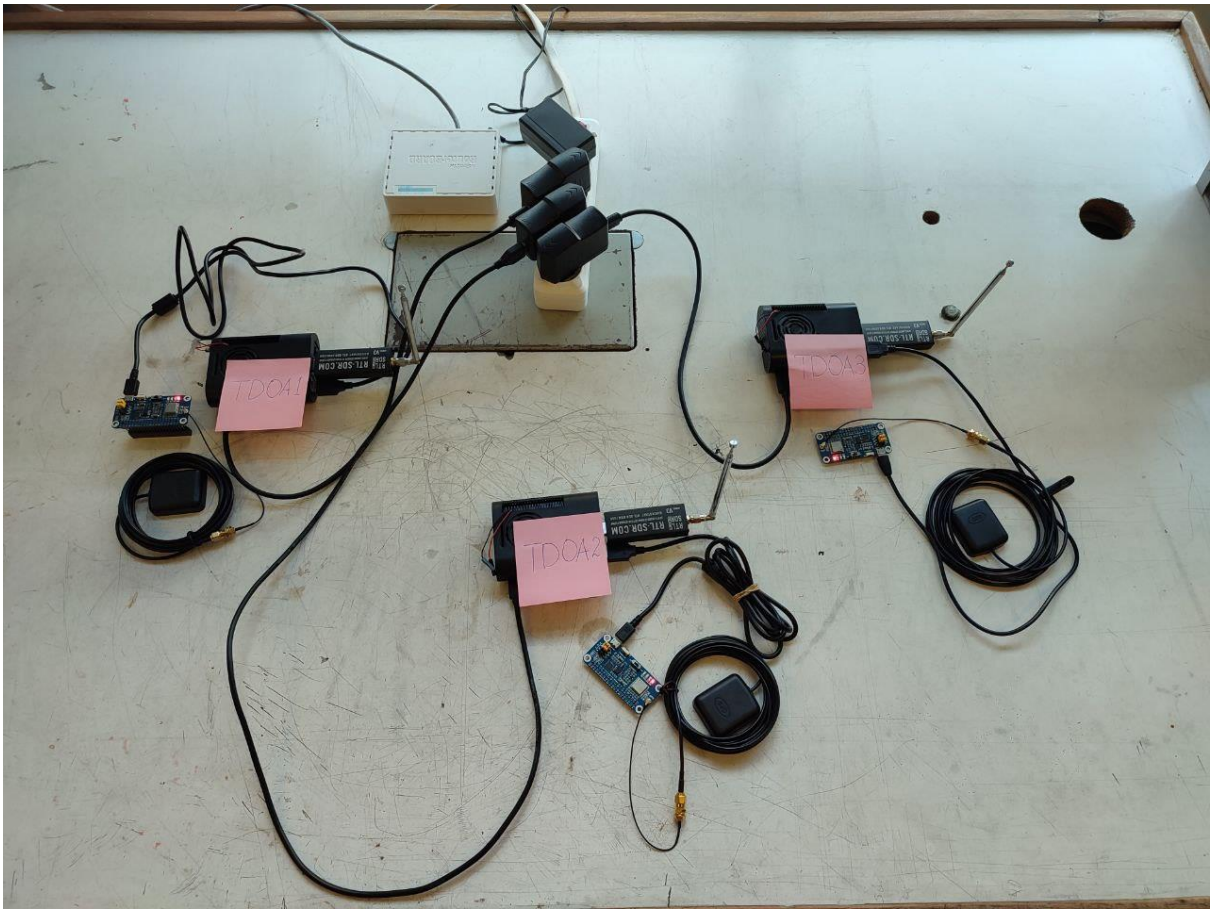

Рисунок 20 – Макет різницево-далекомірної системи

3.1 Аналіз операційних систем для Raspberry Pi

Найперше, що потрібно зробити при розробці програмного забезпечення для побудови різницево-далекомірної системи, так це обрати операційну систему для мікрокомп’ютера, оскільки від вибору операційної системи залежать алгоритм та програмні рішення для реалізації системи.

Мікрокомп’ютери Raspberry Pi підтримують широкий список операційних систем (ОС), серед них[20]:

– Raspbian (або Raspberry Pi OS) – офіційна операційна система, розроблена Raspberry Pi Foundation. Вона базується на Debian Linux і є однією з найпопулярніших ОС для Raspberry Pi.

– Ubuntu – офіційна версія Ubuntu для Raspberry Pi 4, яка підтримується Canonical. Доступні різні редакції, включаючи Ubuntu Server та Ubuntu Desktop.

– Fedora – дистрибутив Linux, який має офіційну версію для Raspberry Pi 4.

– Arch Linux ARM – популярна мінімалістична дистрибутивна Linux-система, яку можна встановити на Raspberry Pi 4.

– Gentoo – ще одна Linux-дистрибутива, що підтримує Raspberry Pi 4.

– Windows 10 IoT Core - спеціалізована версія Windows 10, призначена для вбудованих систем, включаючи Raspberry Pi 4.

– Android – на Raspberry Pi 4 можна встановити деякі версії Android, хоча це зазвичай не є офіційно підтримуваною ОС.

Таке різноманіття операційних систем для побудови реалізації програмного забезпечення для різницево–далекомірної системи надлишкове. Ubuntu має більш широкі можливості, але у цій роботі розширені можливості Ubuntu не потрібні. Найкращим вибором для виконання роботи є Raspbian, оскільки він розробляється та підтримується безпосередньо Raspberry Pi Foundation, творцями самого Raspberry Pi, що гарантує високу сумісність та оптимізацію для апаратного забезпечення Raspberry Pi. Raspbian спеціально налаштований для максимального використання ресурсів Raspberry Pi 4, надає багато корисних інструментів для швидкого виконання задач. Також однією з найвагоміших причин вибору Raspbian як ОС є доступ до великої бібліотеки програмного забезпечення, спеціально адаптованого для Raspberry Pi. Це полегшує знаходження та встановлення необхідних додатків. Вагомою причиною вибору є активна спільнота розробників та ентузіастів Raspberry Pi, що забезпечує підтримку та допомогу.

3.2 Запис операційної системи на microSD для Raspberry Pi

Для використання Raspberry Pi перш за все потрібно записати операційну систему на microSD карту. Саме записати, а не встановити, оскільки Raspberry Pi не має вбудованого накопичувача, замість цього використовуються зйомні microSD накопичувачі. Операційна система — це

набір основних програм і утиліт, які забезпечують роботу комп'ютера. Операційна система поставляється у форматі запакованого архіву образу диска. Для Raspberry Pi існує безліч дистрибутивів, що підтримуються як офіційно (Raspberry Pi Foundation), так і сторонніми компаніями, навіть існують дистрибутиви створені спільнотою. У попередньому розділі ми вияснили, що для задач цієї роботи найбільш доцільно використовувати еталонну операційну систему для Raspberry Pi – Raspbian, що заснована на базі Debian. Debian є дистрибутивом Linux та складається з вільного програмного забезпечення.

Raspbian – це безкоштовна операційна система, оптимізована для апаратного забезпечення Raspberry Pi. Однак Raspbian надає більше, ніж чисту ОС: вона постачається з понад 35 000 пакетів, має попередньо скомпільоване програмне забезпечення в зручному форматі для легкого встановлення на Raspberry Pi[20].

Для запису операційної системи на microSD карту нам знадобиться Card reader – пристрій для читання карт пам'яті. Часто такий пристрій вбудований в корпус персонального комп'ютера чи ноутбука, з якого буде відбуватися запис операційної системи на microSD. Також може знадобитися адаптер для переходу з microSD формату на SD формат, оскільки більшість пристроїв використовує роз'єм саме SD формату[21].

Рисунок 21 – Card reader

Рисунок 22 – microSD карта з SD адаптером

Для початку потрібно форматувати microSD карту, що приведе до видалення всіх даних на цій карті. Якщо на карті є потрібні файли, то скопіюйте їх на інший накопичувач.

Форматувати карту пам’яті можна як за допомогою стандартних засобів Windows, так і за допомогою спеціальних програм. За допомогою стандартних засобів Windows, форматування можна здійснити натиснувши правою кнопкою миші на накопичувач та обрати дію «Форматувати», як на рис. 23.

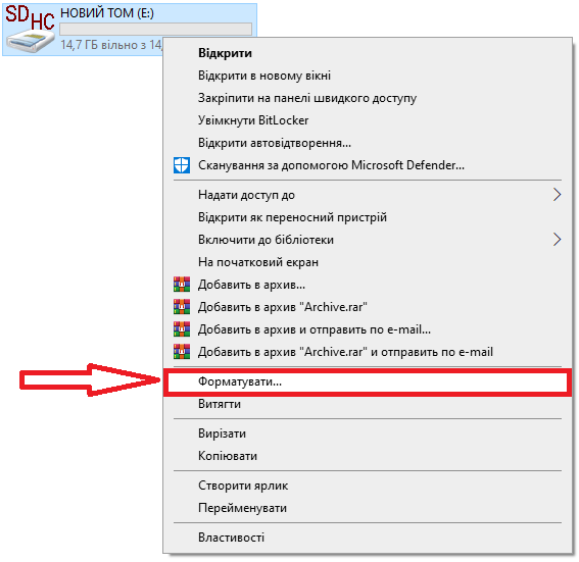

Рисунок 23 – Форматування в файловому провіднику

Більш надійний спосіб форматувати microSD картку засобами Windows можна здійснити за допомогою інструменту «Керування дисками». Для цього натисніть «Пуск» та в пошуку знайдіть «Створити та форматувати розділи жорсткого диска». У відкритому вікні ви побачите усі накопичувачі, що підключені до вашого пристрою. Знайдіть microSD картку та впевніться, що це саме той накопичувач, на який ви хочете записати операційну систему для Raspberry Pi. Підпис на диску має відображатися як «Змінний», а не «Основний». Натисніть правою кнопкою миші на розділ накопичувача та видаліть його. Потрібно видалити всі розділи на накопичувачі, щоб в результаті пам’ять на накопичувачі була не розподілена, як показано на рис. 24.

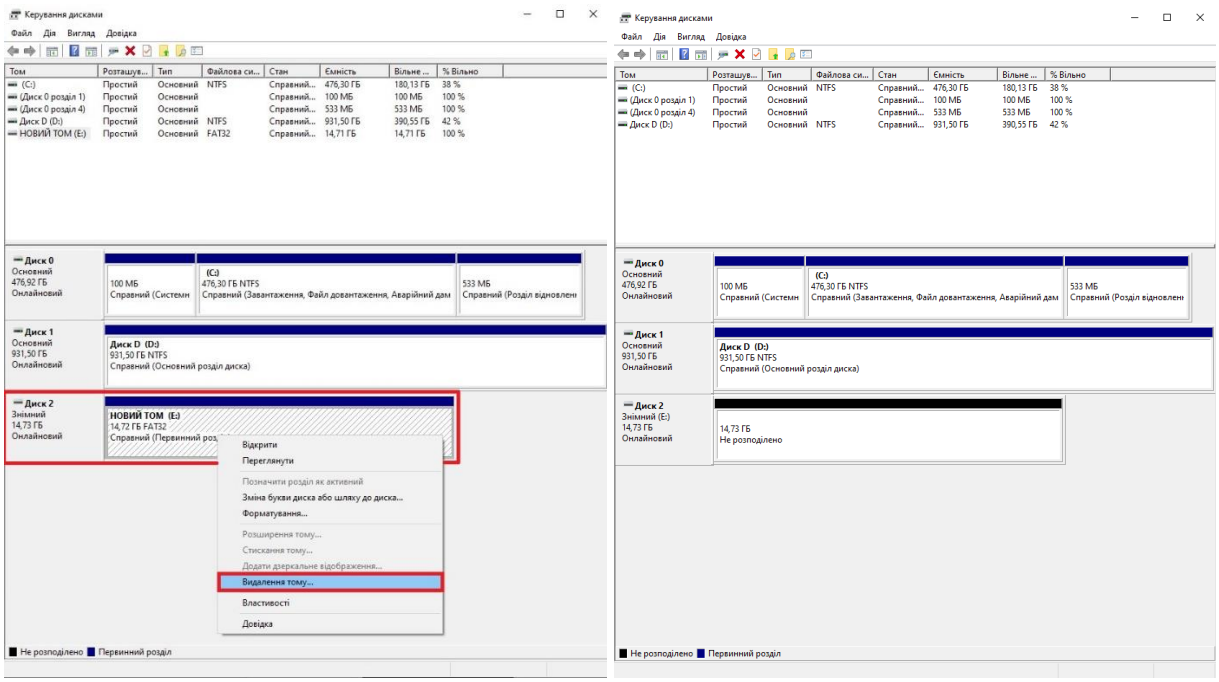

Рисунок 24 – Видалення розділу через «Керування дисками»

Далі потрібно створити простий том, натиснувши правою кнопкою миші на нерозподілену пам’ять накопичувача та залишити налаштування як показано на рис. 25.

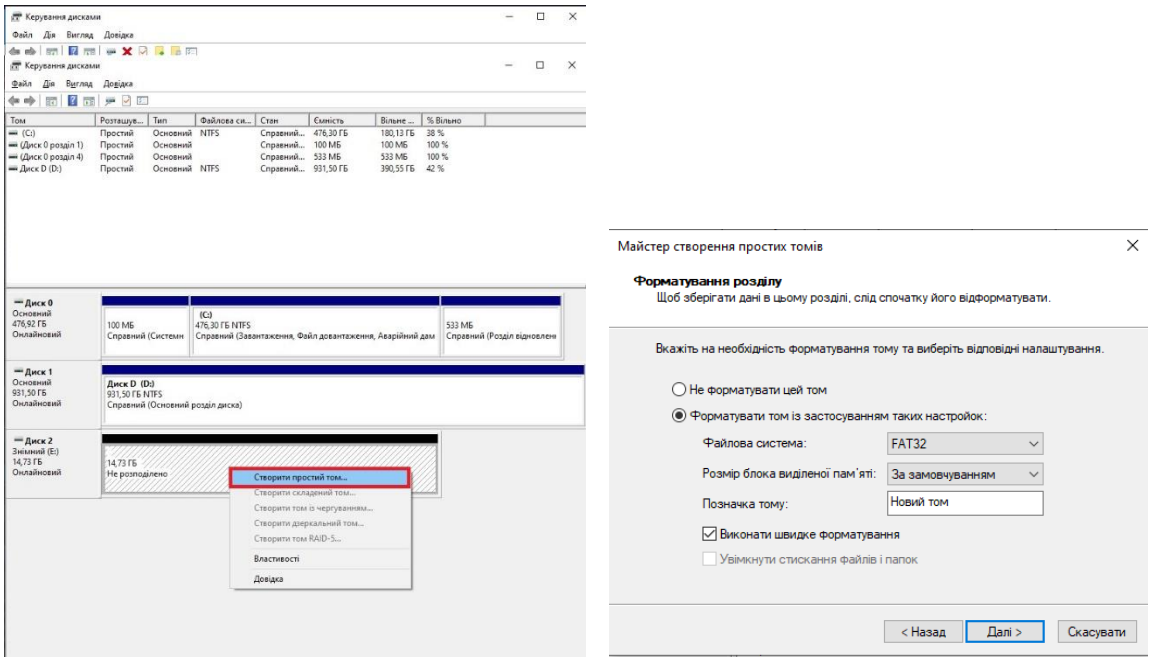

Рисунок 25 – Створення простого тому за допомогою «Керування дисками»

Коли microSD картка відформатована, на неї можна записати операційну систему. Запис операційної системи на microSD картку можна виконати

різними способами та програмами. Найкраще використати спеціальну утиліту від розробника – Raspberry Pi Imager. Утиліта проста у використанні, а завдяки зчитуванню відповідних файлів безпосередньо з сайту розробника, це дозволяє використовувати актуальні версії операційних систем безпечно.

Для запису операційної системи на microSD картку, завантажте утиліту Raspberry Pi Imager з офіційного сайту та відкрийте її. Ви побачите вікно програми рис. 26.

Рисунок 26 – Вікно Raspberry Pi Imager

У відкритому вікні оберіть пристрій, який буде використовувати операційну систему. У нашому випадку це Raspberry Pi 4. Далі потрібно обрати операційну систему, яка буде записана на microSD картку. Є можливість встановити 32-бітну та 64-бітну систему на Raspberry Pi 4. Будемо використовувати 64-бітну систему, оскільки це дозволить більш ефективно використовувати оперативну пам’ять Raspberry, що збільшить швидкість виконання процесів. Серед запропонованих 64-бітних систем бачимо відмінності у вигляді дистрибутивів Bookworm та Bullseye. Bullseye – це стабільна версія, що випущена в 2021 році. Bookworm – це тестова версія, що містить нові інструменти та бібліотеки. У нашому випадку стабільність та оптимізація є перевагою, тому обираємо операційну систему «Raspberry Pi OS (Legacy, 64-bit)». Для цього переходимо в «Raspberry Pi OS (other)» та обираємо відповідну операційну систему.

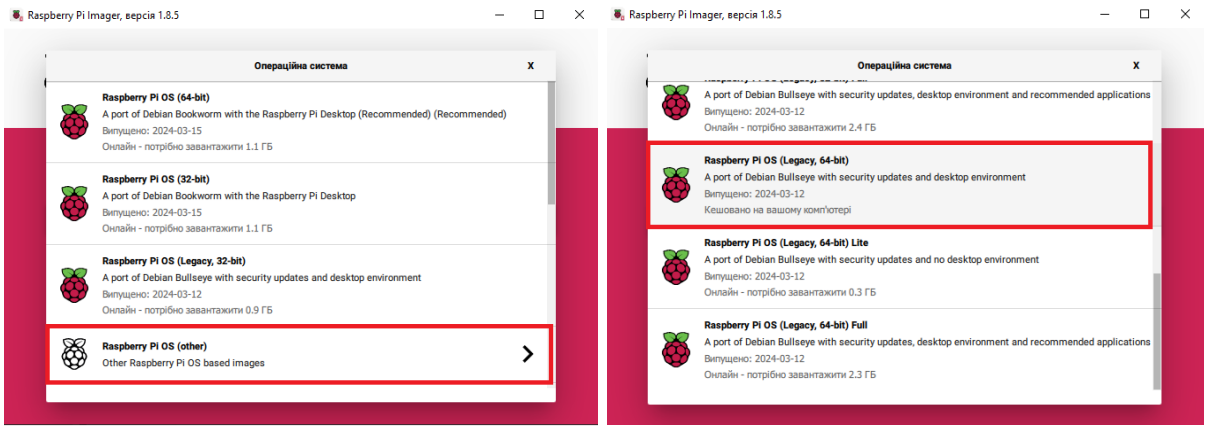

Рисунок 27 – Вибір операційної системи в Raspberry Pi Imager

Обираємо накопичувач, на який буде записано операційну систему і натискаємо «Далі». З’явиться вікно «Використовувати налаштування ОС?». У цьому вікні обираємо «Редагувати налаштування», як показано на рис. 28 та задаємо ім’я хосту, а також ім’я користувача та пароль у вкладці загальних налаштувань. У вкладці сервіси вмикаємо SSH, що дозволить віддалено керувати терміналом мікрокомп’ютера, як показано на рис.29.

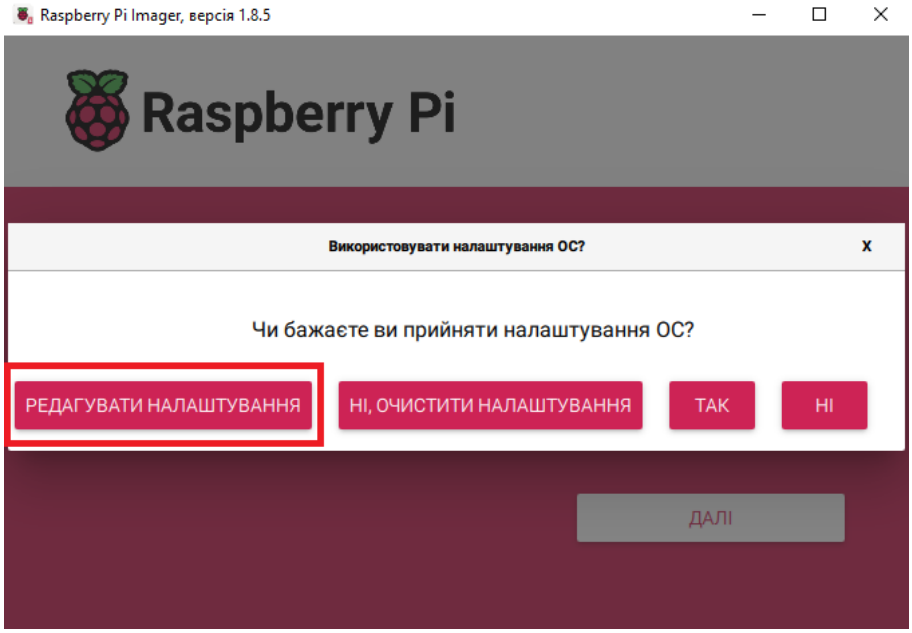

Рисунок 28 – Редагування налаштування операційної системи Raspbian

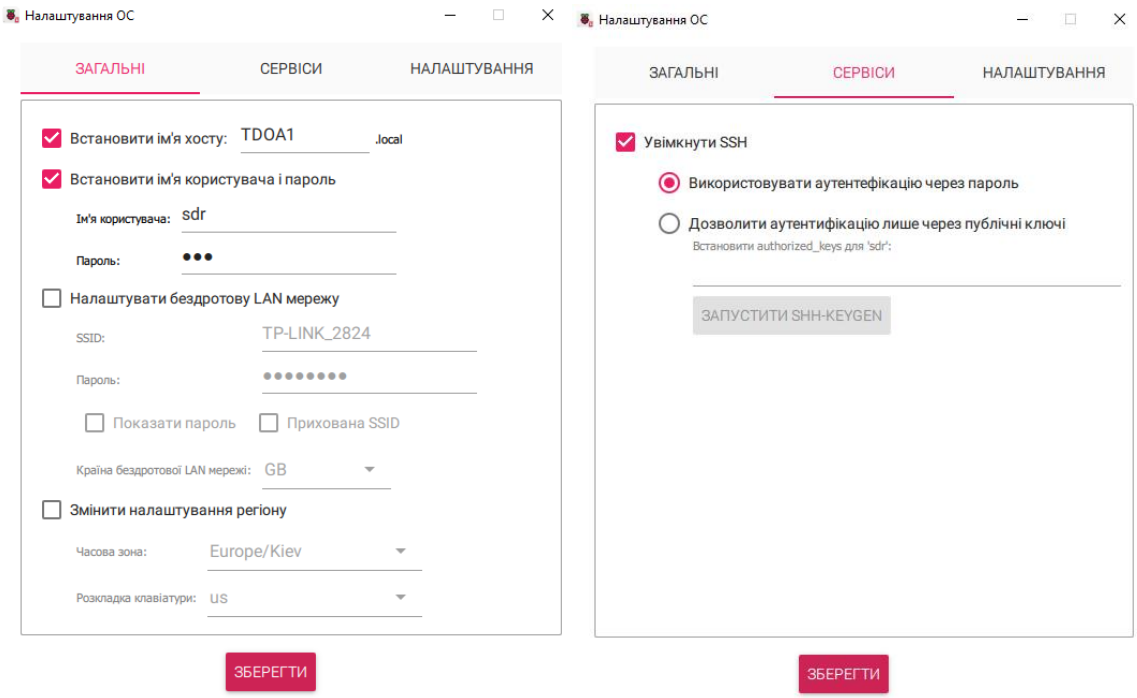

Рисунок 29 – Налаштування операційної системи Raspbian

SSH – це мережевий протокол, що дозволяє віддалено управляти комп'ютером та забезпечує безпечне з'єднання. Для віддаленого підключення потрібна тільки спільна локальна мережа, локальна IP-адреса Raspberry Pi та пристрій з якого буде відбуватися управління терміналом цієї Raspberry Pi.

Існують інші способи віддаленого управління Raspberry Pi. Наприклад, за допомогою віртуальних мережевих обчислень (VNC) можливо керувати одним пристроєм з іншого. VNC покладається на клієнта та сервер. Клієнтом виступає пристрій з якого відбувається керування, а сервер працює на Raspberry Pi. Використання такого способу передає дії клавіатури та миші на сервер, а клієнту відображається екран Raspberry Pi.

Ще одним способом віддаленого керування мікрокомп'ютером є Raspberry Pi Connect, що безпечно надає доступ до терміналу Raspberry Pi без необхідності визначення вашої локальної IP-адреси. Спільний доступ до екрану відбувається віддалено з браузера за допомогою Raspberry Pi Connect.

Загалом є ще багато способів віддаленого керування, але будемо використовувати SSH через простоту налаштування, мале споживання та додаткові функціональні можливості, серед яких є безпечна передача файлів

через SCP (Secure Copy) і SFTP (Secure File Transfer Protocol), що дозволяє легко передавати файли між комп'ютером і Raspberry Pi, що може бути дуже зручним при подальших налаштуваннях.

Зберігаємо налаштування та очікуємо поки йде процес запису операційної системи. Після завершення запису на накопичувач, можна дістати microSD карту. Запис операційної системи на карту завершено. Тепер можна під’єднати microSD карту до Raspberry Pi та продовжити налаштування для реалізації системи виявлення джерел випромінювання.

Не обов’язково вмикати SSH при налаштуванні операційної системи. Увімкнути SSH можна і пізніше, з самого Raspberry Pi за допомогою терміналу[22]:

– Введіть «sudo raspi-config» у вікні терміналу.
– Виберіть Interfacing Options.
– У вікні виберіть SSH.
– Виберіть Yes.
– Виберіть Ok.
– Виберіть Finish.

Також це можна зробити з робочого столу Raspberry Pi:

– У меню Preferences оберіть Raspberry Pi Configuration.
– Перейдіть на вкладку Interfaces .
– Виберіть Enabled поруч із SSH .
– Натисніть OK .

3.3 Опис налаштування програмної частини через термінал для SDR приймача

Після запису операційної системи на microSD картку потрібно встановити драйвери на периферійні пристрої, що використовуються в цій роботі. Почнемо налаштування з встановлення драйверів для коректної роботи

приймача RTL-SDR v3. Для цього потрібно відкрити термінал Raspberry Pi та ввести команду для оновлення репозиторіїв[23]:

sudo apt update

Репозиторії Raspberry Pi – це сховища коду, даних та інших файлів, що пов’язані з розробкою та використанням операційної системи та різних програм чи проектів. Репозиторії містять драйвери, інструменти, та різні версії операційних систем. Репозиторії можуть бути як офіційні, створені розробниками, так і створені спільнотою для реалізації різних проектів.

Команда sudo apt update використовується для завантаження або оновлення локального індексу пакетів і надання системі найновішої інформації про доступні пакети зі сховищ. Оновлюючи індекс пакетів, установку чи оновлення будь-якого програмного забезпечення можна виконувати на основі останньої інформації про пакет.

Команда sudo дозволяє запускати команди з правами адміністратора.

Команда apt – це менеджер пакетів, за допомогою якого можна керувати встановленням, оновленням і видаленням програмного забезпечення.

Команда update – оновлює локальну базу даних доступних пакетів, отримуючи інформацію про останні версії віддалених репозиторіїв.

Далі потрібно встановити деякі утиліти для встановлення драйвера. Потрібно ввести команду в термінал:

sudo apt-get install git

Ця команда запускає менеджер пакетів з правами адміністратора та вказує, що ми хочемо встановити новий пакет за допомогою команди install, а git – це назва пакету, який ми встановлюємо.

Пакет git є популярною системою контролю версій, що дає можливість будь-кому отримати повну копію репозиторію на своєму комп’ютері, а також підтримує роботу з віддаленими репозиторіями, такими як GitHub, Bitbucket та GitLab. Це дозволяє зберігати копії коду та публікувати їх для широкої аудиторії.

Наступною командою вводимо в термінал команду:

sudo apt-get install cmake

Ця команда встановлює пакет CMake, що є системою для керування процесами збирання програмного забезпечення. CMake генерує файли збірки (наприклад, Makefiles для Unix-подібних систем), які потім використовуються для компіляції та збірки програмного забезпечення. CMake приховує низькорівневі деталі збірки за допомогою зрозумілої мови конфігурації. Це дозволяє сфокусуватись на логіці програми, а не на особливостях збірки.

Наступна команда:

sudo apt-get install build-essential

Ця команда встановлює пакет build-essential, що містить набір інструментів, необхідних для компіляції програмного забезпечення з вихідного коду в Linux. Встановлення пакету "build-essential" важливе при роботі з проектами, що використовують CMake. Цей пакет забезпечує наявність необхідних інструментів для успішної збірки проекту на Unix-подібних системах.

Після того, як ми встановили утиліти, що необхідні для збірки та встановлення драйвера, ми можемо отримати необхідний драйвер з сайту GitHub та встановити його.

Перед завантаженням драйвера для приймача RTL-SDR, ми повинні встановити бібліотеку, що дає доступ до роботи з USB пристроями, оскільки SDR приймачі підключаються до комп’ютера за допомогою USB порту.

Вводимо команду для встановлення бібліотеки libusb-1.0-0-dev, що дає можливість працювати з USB пристроями, за допомогою команди:

sudo apt install libusb-1.0-0-dev

Коли відповідні бібліотеки встановлені, можемо завантажити копію Git репозиторію на комп’ютер, ввівши в термінал команду:

git clone https://gitea.osmocom.org/sdr/rtl-sdr.git

Після виконання цієї команди буде створено папку rtl-sdr у поточному робочому каталозі. В цій папці буде міститися весь вміст віддаленого репозиторію з драйвером для приймача SDR.

Далі відкриємо створену папку, змінивши поточний робочий каталог на rtl-sdr. Це можна зробити за допомогою команди:

cd rtl-sdr/

Команда cd означає "change directory" (змінити директорію), а rtl-sdr/ – це назва каталогу (папки), в який ми переходимо.

Наступним кроком створимо папку під назвою "build" за допомогою команди mkdir, що означає "make directory" (створити каталог):

mkdir build

Перейдемо в створений каталог build:

cd build

Далі використовуємо команду для запуску раніше встановленого інструменту CMake, що дозволить підготувати середовище для компіляції проекту RTL-SDR:

cmake ../ -DINSTALL_UDEV_RULES=ON DDETACH_KERNEL_DRIVER=ON

Ця команда вмикає встановлення правил udev (Unified Device Model) під час збірки. Правила udev дозволяють коректно ідентифікувати та використовувати RTL-SDR-пристрої.

Після виконання цієї команди в каталозі "build" буде згенеровано всі необхідні файли збірки (Makefiles, проектні файли тощо), готові для наступного етапу - компіляції.

DDETACH_KERNEL_DRIVER=ON - цей параметр вказує CMake, що необхідно від'єднати вбудований драйвер ядра для пристрою. Це потрібно, щоб запобігти конфліктам між вбудованим драйвером та спеціалізованим драйвером, який буде використовуватися в проекті.

Для компіляції потрібно ввести команду:

make

Після введення даної команди запускається зчитування файлів для збірки (Makefiles, проектні файли тощо) і відбувається автоматична компіляція вихідного коду зі створенням виконуваних файлів. Цей процес

може зайняти деякий час, а після виконання в каталозі build будуть знаходитися скомпільовані файли.

Ці файли потрібно встановити на Raspberry Pi. Для цього введіть в термінал команду:

sudo make install

Після виконання цієї команди, файли будуть встановлені в правильні системні каталоги, де вони будуть доступні для використання. Коли команда виконана, ми отримуємо повністю встановлене програмне забезпечення RTL-SDR на своєму комп'ютері. Тепер його можна використовувати в різних застосунках, що працюють з RTL-SDR-пристроями.

Далі потрібно оновити кеш динамічних бібліотек, оскільки ми встановили нові бібліотеки RTL-SDR. Після встановлення нових бібліотек, необхідно оновити кеш, щоб ми могли переконатися, що бібліотеки будуть коректно завантажуватись і використовуватись програмами, які з ними працюють. Для цього потрібно ввести команду:

sudo ldconfig

Після всіх цих операцій потрібно перезавантажити Raspberry Pi для того, щоб після встановлення драйверів та внесення системних змін ми змогли завершити процес встановлення та звільнити пам’ять та ресурси, що були застосовані під час процесу встановлення драйверів. Для перезавантаження Raspberry Pi потрібно ввести в термінал команду:

sudo reboot

Коли мікрокомп’ютер перезавантажився, можемо протестувати чи успішно встановився драйвер для RTL-SDR приймача. Для цього введемо команду:

rtl_test

Ця команда – це утиліта з пакету інструментів RTL-SDR. Вона дозволить перевірити правильність підключення та функціонування RTL-SDR приймача. Ця команда ініціалізує підключення до RTL-SDR приймача, виводить інформацію про виявлення приймача, проводить базові тести для перевірки

правильної роботи пристрою, а також виводить результати тестування. Після виконання цієї команди, в терміналі має відобразитися подібна інформація, як на рис. 30.

```
sdr@TDOA1:~ $ rtl_test
Found 1 device(s):
  0:  Generic, RTL2832U, SN: 77771111153705700

Using device 0: Generic RTL2832U
Detached kernel driver
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7
[R82XX] PLL not locked!
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
```

Рисунок 30 – Вигляд терміналу після введення команди rtl_test

Для того щоб завершити тестування потрібно натиснути комбінацію клавіш Ctrl + C. Якщо тестування пройшло успішно, то драйвер для RTL-SDR встановлений правильно і приймач готовий до експлуатації.

3.4 Опис налаштування програмної частини через термінал для GPS модуля L76X GPS HAT

Коли приймач налаштований і здатен приймати сигнали від джерел випромінювання, потрібно приступити до налаштування модулю прийому GPS координат, який буде використаний для синхронізації між усіма мікрокомп’ютерами системи визначення місцезнаходження джерел випромінювання.

Сам модуль L76X GPS HAT має вигляд як на рис.31.

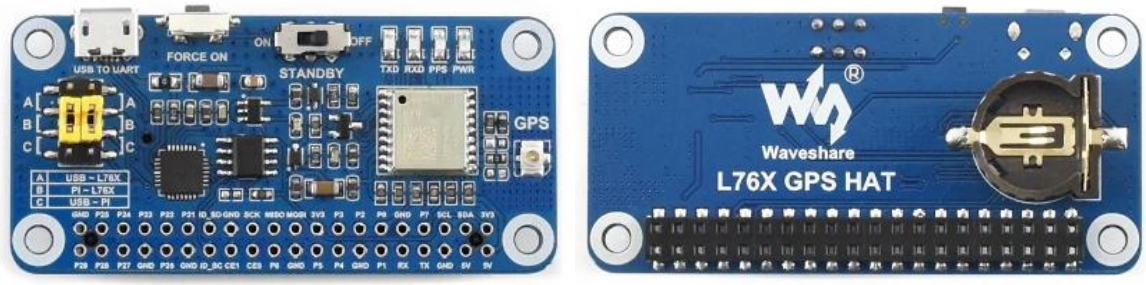

Рисунок 31 – L76X GPS HAT

Принцип налаштування модуля GPS схожий з налаштуваннями SDR приймача. Для початку потрібно ввести команду в термінал Raspberry Pi:

sudo apt-get update

Ця команда оновлює локальну базу даних пакетів, завантажуючи інформацію про останні версії пакетів з репозиторіїв, вказаних у файлах конфігурації системи управління пакетами.

Наступна команда встановлює бібліотеку Python для роботи з послідовними портами:

sudo apt-get install python3-serial

Бібліотека python3-serial є дуже корисною при роботі з різноманітними пристроями, що використовують послідовні інтерфейси для передачі даних, включаючи GPS модулі. Вона дозволяє легко надсилати та отримувати дані через послідовні порти з використанням простих функцій Python.

Далі встановимо пакети, які необхідні для роботи з модулем GPS. Для цього в термінал потрібно ввести команду:

sudo apt-get install gpsd gpsd-clients python3-gps

Розглянемо функції встановлених пакетів. Gpsd – це демон, який обробляє дані, отримані з GPS-пристрою, і надає уніфікований інтерфейс для доступу до цих даних для різних клієнтів. Термін «демон» у контексті операційних систем Linux означає фонову службу або процес, який працює в режимі очікування та виконує певні системні завдання.

Gpsd-clients – це набір клієнтських програм, які можуть взаємодіяти з демоном gpsd для отримання даних з GPS-пристрою.

Python3-gps – це бібліотека Python, яка дозволяє отримувати дані з GPS-пристрою через демона gpsd. Це дає можливість інтегрувати GPS-функціональність безпосередньо у Python-програми. Встановивши ці пакети, ми отримаємо всі необхідні інструменти для налаштування та використання GPS модуля.

Тепер можна приступити до апаратного підключення L76X GPS HAT до Raspberry Pi. Перш за все потрібно підключити антену до модуля GPS, а інший

кінець з приймальної антени потрібно розмістити там, де можна безпосередньо бачити небо без усіляких перешкод, наприклад дах, гілки дерев тощо. Далі потрібно підключити USB до Raspberry Pi, а іншою стороною кабелю з micro USB потрібно підключитися до модуля L76X GPS HAT. На модулі наявні жовті перемички A, B та C. Функція перемичок полягає в тому, щоб перемикати пристрій послідовного порту. Коли перемичка підключена до A, керування L76X буде відбуватися через USB-UART. Коли перемичка підключена до B, то L76X буде підключено до Raspberry Pi через GPIO. Коли перемичка підключена до C, то надається доступ до Raspberry Pi через USB-UART. Оскільки ми будемо підключатися через USB, то в нашому випадку потрібно підключити жовті перемички в положення A. Результат апаратного підключення матиме вигляд як на рис. 32.

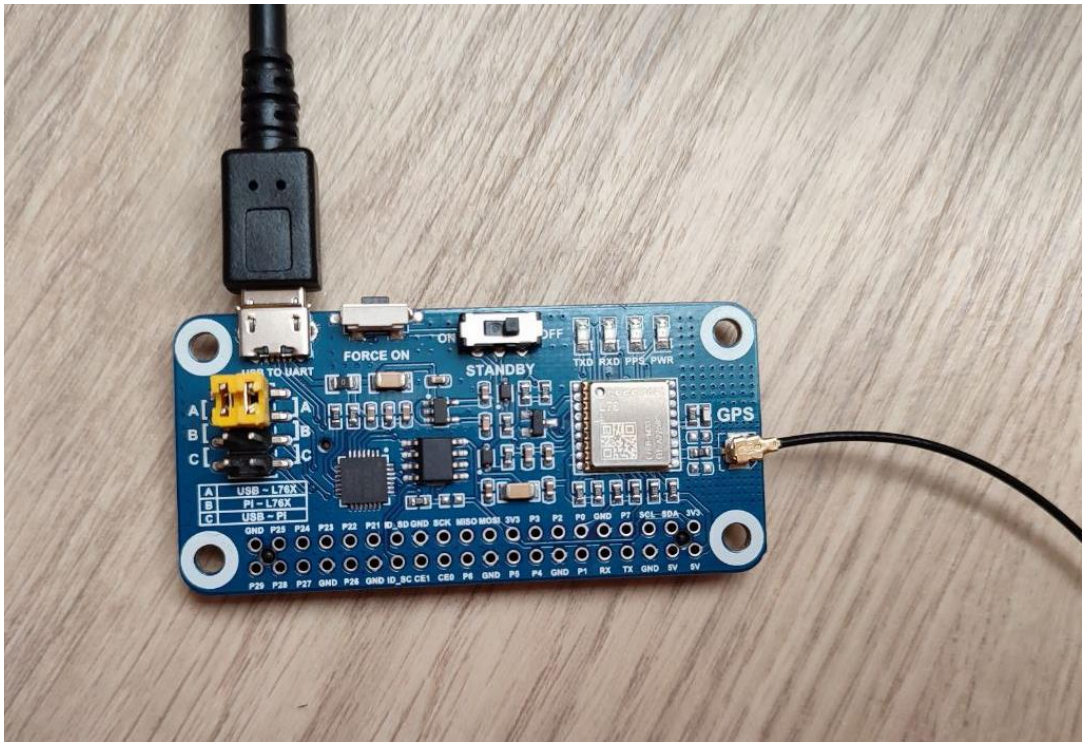

Рисунок 32 – Апаратне підключення L76X GPS HAT

Також на платі модуля знаходиться перемикач STANDBY. Функція STANDBY полягає в тому, що при положенні ON, модуль буде знаходитися в режимі очікування, який є режимом низького енергоспоживання. У режимі очікування модуль припиняє супутниковий пошук і навігацію, і інформація про позиціонування не виводиться, але до неї можна отримати доступ через

команди PMTK або будь-які інші дані. Цей режим роботи не потрібен, тому перемикач STANDBY повинен бути в положенні OFF.

Коли необхідні пакети встановлені, а модуль GPS правильно апаратно підключений, можемо завантажити архів з файлами для GPS-модуля L76X GPS HAT. Для цього потрібно ввести команду:

wget https://files.waveshare.com/upload/f/f4/L76X_GPS_HAT_Code.zip

Ця команда завантажить вказаний ZIP-файл з URL-адреси офіційного сайту модуля L76X GPS HAT та збереже його в поточній робочій директорії.

Після завантаження архіву, потрібно дістати файли з архіву, щоб отримати доступ до вихідного коду та файлів в цьому архіві. Для цього потрібно ввести команду:

unzip L76X_GPS_HAT_Code.zip

Після того, як ми дістали файли з архіву, потрібно встановити повні права доступу для всіх файлів та директорій в папці L76X_GPS_HAT_Code, яку ми розархівували. Для цього потрібно ввести команду:

sudo chmod 777 -R L76X_GPS_HAT_Code

Тепер перейдемо в директорію RaspberryPi, а потім в директорію python. Зробити це можливо за послідовного вводу таких команд:

cd L76X_GPS_HAT_Code/RaspberryPi

cd python

Перейшовши в директорію python, потрібно встановити бібліотеку pynmeagps за допомогою pip3. Для цього в терміналі потрібно ввести команду:

sudo pip3 install pynmeagps

Бібліотека pynmeagps спрощує роботу з NMEA-даними, які зазвичай генеруються GPS-модулями. Після встановлення цієї бібліотеки ми зможемо використовувати Python-скрипти, які зчитуватимуть та обробляють дані з нашого GPS-модуля. Це надасть доступ до таких важливих даних, як географічні координати, швидкість, час, кількість супутників тощо.

Наступною командою потрібно ввести:

sudo pip3 install gps3

Бібліотека gps3 надає простий інтерфейс для читання та обробки NMEA-даних, отриманих від GPS-пристроїв. Поєднуючи gps3 та pynmeagps, ми маємо набір інструментів для взаємодії з вашим GPS-модулем L76X. Gps3 забезпечує основну функціональність, а pynmeagps додає можливість глибшої обробки даних.

Перед запуском скрипту для отримання GPS координат, потрібно змінити файл конфігурації, що знаходиться в теці python. Змінити його потрібно тому, що за замовчуванням передбачено підключення через послідовні порти GPIO, але ми будемо використовувати підключення через USB для більшої зручності, оскільки корпус для Raspberry Pi не завжди дозволяє зручно підключити модулі через GPIO.

Шлях до конфігураційного файлу виглядає наступним чином: /home/sdr/L76X_GPS_HAT_Code/RaspberryPi/python. Конфігураційний файл має назву config.py, а відкрити його можливо віддалено, оскільки ми увімкнули SSH, що включає в себе SFTP – надійний та безпечний метод для передачі файлів між комп’ютерами за допомогою SSH. Найбільш зручний та наглядний спосіб відкрити цей конфігураційний файл буде за допомогою програми WinSCP. Для цього потрібно завантажити програму з офіційного сайту та підключитися до Raspberry Pi через протокол SSH. Для цього потрібно ввести IP-адресу, а також логін та пароль для входу в систему. Після того, як ми ввели шлях до конфігураційного файлу, вікно WinSCP матиме вигляд як на рисунку 33[24].

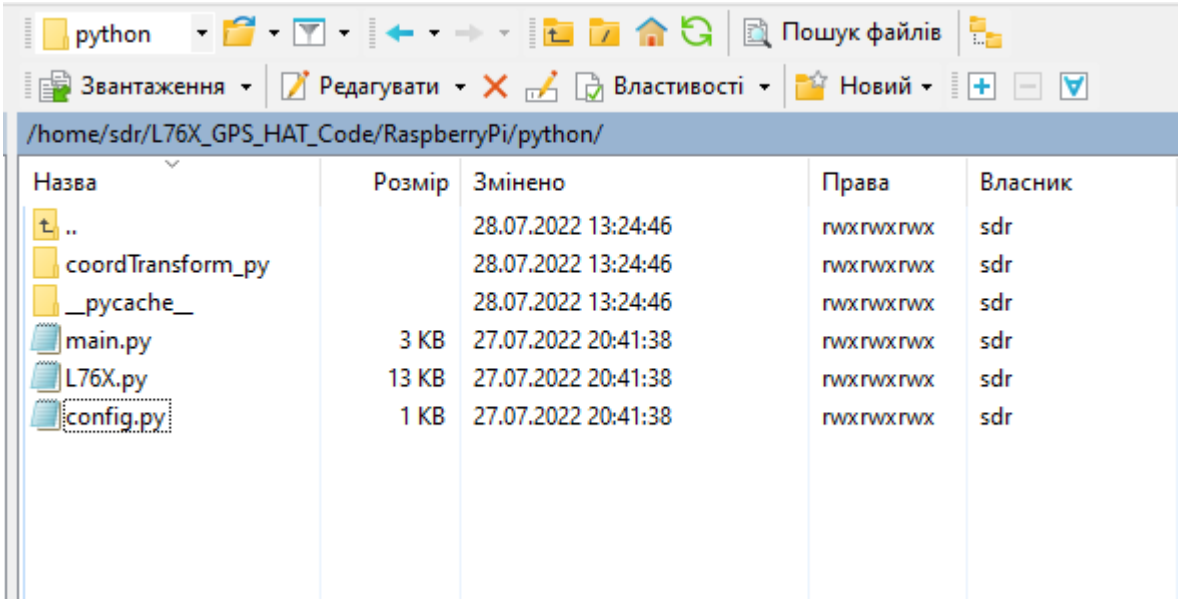

Рисунок 33 – Розташування конфігураційного файлу L76X GPS HAT

Потрібно відкрити конфігураційний файл в текстовому документі та замінити деякі рядки, показані на рис. 34 та рис. 35. Ці рядки встановлюють параметри послідовного UART інтерфейсу, який використовується для підключення GPS модуля до Raspberry Pi. Нам потрібно змінити шлях до фізичного послідовного порту Raspberry Pi, який використовується для підключення GPS-модуля. Зараз шлях /dev/ttyS0 відповідає послідовному UART порту, а для підключення через USB потрібно змінити цей рядок на /dev/ttyUSB0.

/home/sdr/L76X_GPS_HAT_Code/RaspberryPi/python/config.py – 192.168.0.104 – Редактор – WinSCP

<Кодування> Колір

```
#!/usr/bin/python
# -*- coding:utf-8 -*-
import serial
import RPi.GPIO as GPIO

Temp = '0123456789ABCDEF*'

class config(object):
    FORCE  = 17
    STANDBY= 4
    def __init__(ser, Baudrate = 9600):
        ser.serial = serial.Serial("/dev/ttyS0",Baudrate)
        GPIO.setmode(GPIO.BCM)
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(ser.FORCE, GPIO.IN)
        GPIO.setup(ser.STANDBY, GPIO.OUT)
        GPIO.output(ser.STANDBY, GPIO.HIGH)

    def Uart_SendByte(ser, value):
        ser.serial.write(value)

    def Uart_SendString(ser, value):
        ser.serial.write(value)

    def Uart_ReceiveByte(ser):
        return ser.serial.read(1)

    def Uart_ReceiveString(ser, value):
        data = ser.serial.read(value)
        return data

    def Uart_Set_Baudrate(ser, Baudrate):
        ser.serial = serial.Serial("/dev/ttyS0",Baudrate)

    def close(ser):
        GPIO.cleanup()
        ser.serial.close()
```

Рисунок 34 – Конфігураційний файл з використанням UART порту

```
#!/usr/bin/python
# -*- coding:utf-8 -*-
import serial
import RPi.GPIO as GPIO

Temp = '0123456789ABCDEF*'

class config(object):
    FORCE  = 17
    STANDBY= 4
    def __init__(ser, Baudrate = 9600):
        ser.serial = serial.Serial("/dev/ttyUSB0",Baudrate)
        GPIO.setmode(GPIO.BCM)
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(ser.FORCE, GPIO.IN)
        GPIO.setup(ser.STANDBY, GPIO.OUT)
        GPIO.output(ser.STANDBY, GPIO.HIGH)

    def Uart_SendByte(ser, value):
        ser.serial.write(value)

    def Uart_SendString(ser, value):
        ser.serial.write(value)

    def Uart_ReceiveByte(ser):
        return ser.serial.read(1)

    def Uart_ReceiveString(ser, value):
        data = ser.serial.read(value)
        return data

    def Uart_Set_Baudrate(ser, Baudrate):
        ser.serial = serial.Serial("/dev/ttyUSB0",Baudrate)

    def close(ser):
        GPIO.cleanup()
        ser.serial.close()
```

Рисунок 35 – Конфігураційний файл з використанням USB порту

Після того, як зміни в файл config.py були внесені, потрібно зберегти ці зміни та закрити конфігураційний файл. Після цього потрібно перезавантажити мікрокомп’ютер, щоб завершити встановлення драйверу для GPS модуля. Для цього потрібно ввести команду:

sudo reboot

Після перезавантаження можемо перевіримо чи працює модуль і чи отримує він данні з супутників. Це можна зробити за допомогою популярної термінальної програми Minicom, яка використовується для налагодження та взаємодії з різними послідовними пристроями, такими як модеми, GPS-модулі тощо. Під час виконання програми, ми отримаємо необроблені дані та зможемо дізнатися чи приймає модуль GPS ці дані без помилок. Для того, щоб встановити помічника з налагодження minicom, потрібно ввести в термінал команду:

sudo apt-get install minicom

Після виконання команди, буде встановлено пакет minicom. Для запуску термінальної програми Minicom та встановлення з'єднання з пристроєм, підключеним до USB-послідовного порту комп'ютера потрібно ввести команду:

sudo minicom -D /dev/ttyUSB0 -b 9600

Ця команда показує, що Minicom повинен використовувати пристрій /dev/ttyUSB0 для послідовного зв'язку. Також ця команда встановлює швидкість передачі даних 9600 біт/с. Це стандартна швидкість для багатьох послідовних пристроїв. Результатом виконання програми має бути відображення необроблених даних з супутників, як показано на рис. 36.

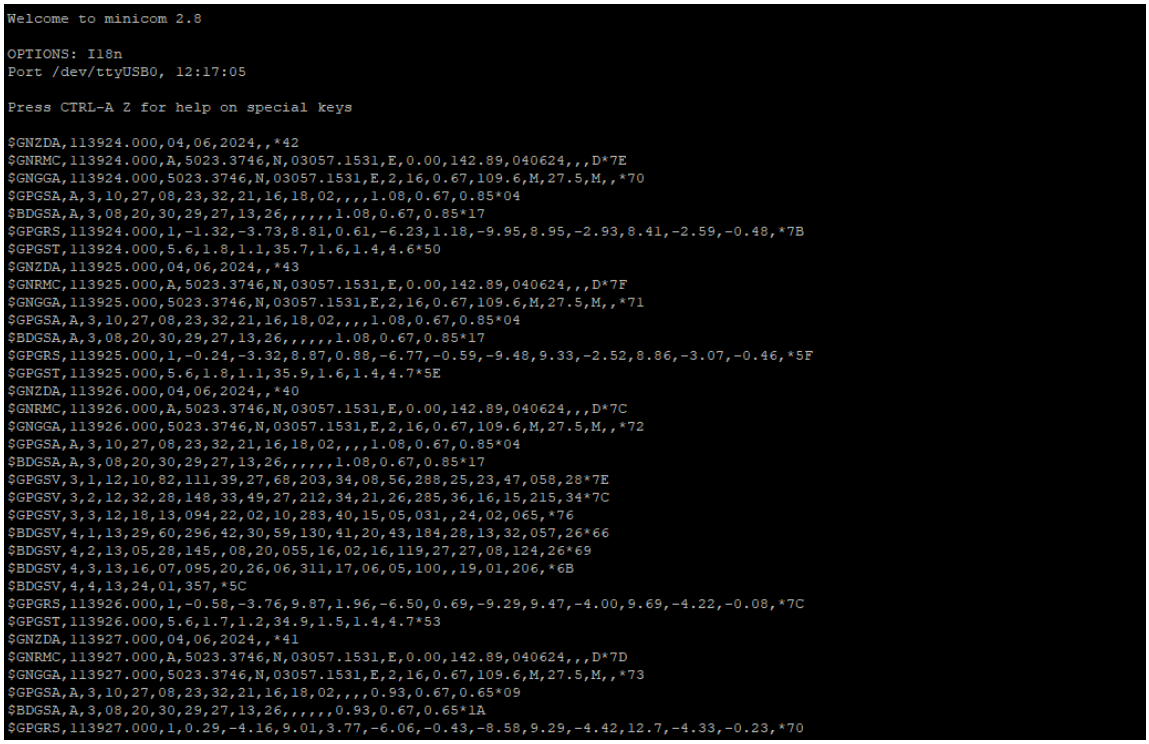

Рисунок 36 – Вигляд програми Minicom Debug

Для виходу з програми потрібно натиснути поєднання клавіш Ctrl + A, після чого потрібно натиснути клавішу X та у вікні «Leave without reset» обрати Yes та натиснути Enter.

Після того, як ми впевнилися в правильній роботі модуля, можна запустити скрипт для отримання координат, що міститься в файлі main.py.

Щоб запустити main.py, потрібно знаходитися в правильній директорії, де знаходиться цей файл. Для того щоб перейти в директорію з файлом main.py, потрібно ввести такі команди в термінал:

cd L76X_GPS_HAT_Code/RaspberryPi

cd python

Для запуску файлу main.py, що містить скрипт для отримання даних з GPS модуля, потрібно ввести команду:

sudo python3 main.py

Після виконання цієї команди ми отримуємо GPS координати нашого модуля і відповідно, координати мікрокомп'ютера. Очікуваний результат матиме наступний вигляд, як на рис. 37.

```
sdr@TDOA1:~ $ cd L76X_GPS_HAT_Code/RaspberryPi
sdr@TDOA1:~/L76X_GPS_HAT_Code/RaspberryPi $ cd python
sdr@TDOA1:~/L76X_GPS_HAT_Code/RaspberryPi/python $ sudo python3 main.py
gps device make wgs84 coordinate
wgs84 coordinate is for OpenStreetMap
gcj02 coordinate is for amap or google map
bd09 coordinate is for baidu map
 Please press Ctrl+c if want to exit
altitude       =  n/a Mse put the antenna outdoors and wait a moment
wgs84 lon,lat  =  30.95263 , 50.389621667
google lon.lat = 50.389621667,30.952630000
amap lon.lat   = 30.952630000,50.389621667
bd09 lon,lat   = 30.958984535,50.395698885
speed          =  0.0 KM/H
Update after 4 secondss
```

Рисунок 37 – Результат виконання main.py

Для першого позиціонування модуля потрібно приблизно 35 секунд. Перша частина виведеної інформації — вихідні дані модуля. Широта та довгота – це вихідні широта та довгота. Система координат Baidu ставить довготу наперед, тому що отримані необроблені дані GPS – це необроблені географічні координати, які потрібно додати до певного алгоритму, щоб бути точними. Різні карти використовують різні алгоритми. Такі ж координати є в Google і Baidu, але обробка різна. Тому в різних алгоритмах може бути різна точність, а широта і довгота можуть бути змінені порядком відображення.

Отже, ми отримали координати з GPS модуля, що означає правильне програмне та апаратне налаштування модуля L76X GPS HAT.

3.5 Підключення Raspberry Pi до бездротової локальної мережі

Для більш зручного та мобільного з’єднання між Raspberry Pi можна використати з’єднання по Wi-Fi. Оскільки дальність з’єднання за допомогою стандартних Wi-Fi сигналів на частоті 2.4 ГГц на відкритій місцевості близько 100 метрів, то для реалізації рознесеної системи визначення місцеположення джерела сигналу потрібно буде використати підсилювачі сингалу або ретранслятори сигналу.

Для підключення Raspberry Pi до бездротової мережі потрібно перевірити чи встановлені необхідні пакети підключення до Wi-Fi мережі та встановити їх:

sudo apt-get update

sudo apt-get install -y wireless-tools wpasupplicant

Коли відповідні пакети встановлені, потрібно перейти до директорії конфігураційного файлу Wi-Fi:

cd /etc/wpa_supplicant/

І відповідно відкрити цей конфігураційний файл для редагування:

sudo nano wpa_supplicant.conf

У цьому конфігураційному файлі потрібно змінити скорочення країни на необхідну, також змінити назву Wi-Fi мережі та ввести пароль від цієї мережі. На прикладі підключення до Wi-Fi мережі з назвою «Drone2» та паролем «123456789» показаний вигляд конфігураційного файлу на рис. 38.

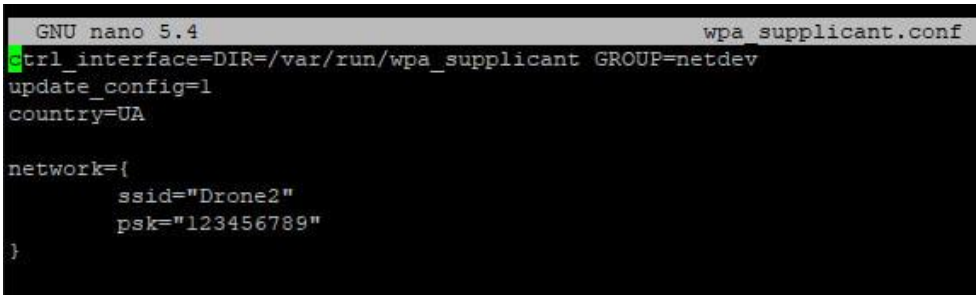

Рисунок 38 – Вигляд конфігураційного файлу для підключення до мережі Drone2

Після внесення змін в конфігураційний файл, його потрібно зберегти та закрити.

Для автоматичного підключення до Wi-Fi мережі при запуску системи потрібно ввести команду:

auto wlan0

Далі потрібно налаштувати інтерфейс wlan0 для автоматичного отримання IP-адреси для Raspberry Pi, а також вказати місце збереження конфігураційного файлу з налаштуваннями мережі Wi-Fi. Для цього потрібно ввести команду:

iface wlan0 inet dhcp

wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf

Після виконання команд на всіх мікрокомп'ютерах системи, Raspberry Pi будуть підключені до бездротової мережі, що забезпечить більшу мобільність системи без втрати її функціональності [25].

3.6 Налаштування системи для підключення між Raspberry Pi без вводу паролю

Усі вищеописані дії для налаштування SDR приймача та GPS модуля потрібно повторити з кожним мікрокомп'ютером, що використовується в системі. Після чого ми обираємо будь-який Raspberry Pi, як основний, на якому будуть зберігатися сигнали для подальшої обробки. Потрібно знайти шлях, як швидко отримувати сигнали з усіх мікрокомп'ютерів одночасно та так само швидко передавати ці сигнали на основний мікрокомп'ютер. Для цього найкраще використати bash-файли. Більш повна інформація про створення bash-файлів, їх наповнення та функції буде надана в наступному розділі. Для того, щоб використати bash-файли для автоматичної передачі записаних сигналів на основний Raspberry Pi, потрібно через протокол SSH отримати доступ до файлів на інших Raspberry Pi, що використовуються в системі, скопіювати ці файли на основний мікрокомп'ютер та обробити ці сигнали. Це

можливо реалізувати, але є одна проблема і полягає вона в тому, що копіювання файлів за замовчуванням передбачає введення паролю для входу в систему мікрокомп’ютера, з якого копіюють файл. Таким чином, автоматичне копіювання неможливе, оскільки при запиті на ввід паролю, мікрокомп’ютер очікує введення з клавіатури.

Рішенням цієї проблеми можуть виступати SSH ключі. Далі буде описаний алгоритм налаштування автентифікації на основі ключа SSH, яка дає змогу підключатися до віддаленого сервера без введення пароля користувача.

Перед початком створення ключів, потрібно запустити усі Raspberry Pi, що використовуються в системі, та обрати один основний, на якому буде згенеровано id_rsa ключ.

RSA ключі – це тип криптографічних ключів, що використовуються для шифрування, розшифрування та цифрового підпису даних. Вони застосовуються в багатьох протоколах безпеки, наприклад таких, як SSH. Ключі SSH, зокрема RSA ключі, використовуються для аутентифікації між комп'ютерами при встановленні SSH-з'єднань. Ці ключі допомагають уникнути необхідності вводити пароль кожен раз при підключенні[26].

Далі потрібно дізнатися IP-адресу кожного мікрокомп’ютера, у моєму випадку IP-адреси наступні:

- TDOA1 192.168.88.235
- TDOA2 192.168.88.233
- TDOA3 192.168.88.237

Для зручності були пронумеровані відповідні мікрокомп’ютери, а основним обраний TDOA1. Для створення RSA ключа потрібно ввести в терміналі команду:

ssh-keygen

Після виконання команди, в терміналі з’явиться запит на зміну розташування ключа та введення паролю для цього ключа. Директорію не змінюємо, а в полі введення паролю залишаємо пустий запит і натискаємо Enter. Буде згенеровано 2 ключа, відкритий та закритий. Закритий (приватний)

ключ зберігається локально на основному комп'ютері і ніколи не повинен бути розкритий. Відкритий ключ відповідає приватному ключу, але призначений для поширення. При успішній генерації ключа, термінал матиме вигляд, як на рис. 39.

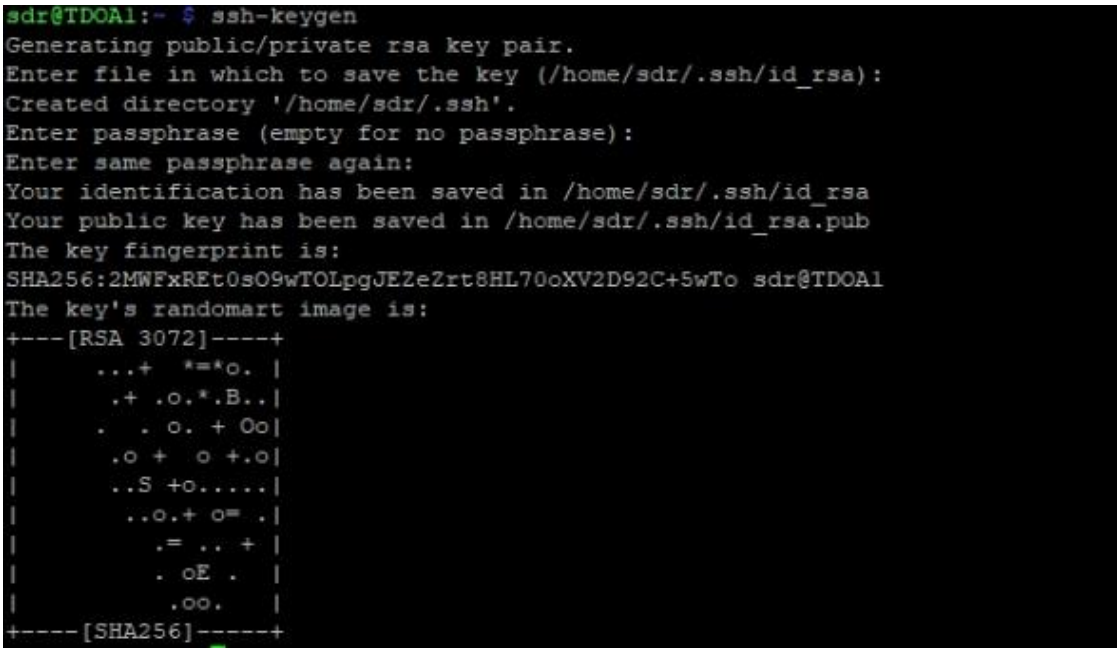

Рисунок 39 – Генерація SSH ключів через термінал

Коли на основному Raspberry Pi згенеровано ключі, потрібно скопіювати відкриті ключі для підключення без пароля на інші мікрокомп’ютери системи. Щоб скопіювати відкритий ключ id_rsa.pub на інші мікрокомп’ютери, потрібно ввести команду:

ssh-copy-id

Ця команда скопіює відкритий ключ на мікрокомп’ютер, яким можна керувати без введення паролю. До цієї команди потрібно додати ім’я та IP-адресу віддаленої машини, до якої ми хочемо отримати доступ. Виглядати команда має наступним чином, як на прикладі копіювання ключа для мікрокомп’ютера TDOA2:

ssh-copy-id sdr@192.168.88.233

Після виконання цієї команди в терміналі з’явиться запит на підтвердження копіювання ключа, вводимо в термінал «yes» та натискаємо Enter, далі з’явиться запит на введення паролю користувача віддаленої

машини, вводимо пароль та натискаємо Enter . В терміналі це матиме вигляд, як показано на рис. 40.

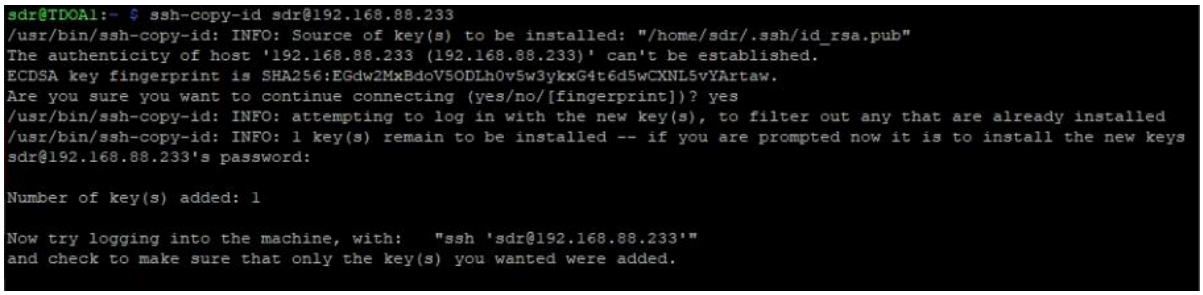

Рисунок 40 – Копіювання відкритого ключа на TDOA2

Тепер можна підключитися до мікрокомп’ютера TDOA2 без введення паролю. Перевірити це можливо за допомогою команди:

ssh

Так само, до цієї команди потрібно додати ім’я та IP-адресу віддаленого мікрокомп’ютера. На прикладі підключення до TDOA2, термінальна команда матиме наступний вигляд, зображений на рис. 41.

Рисунок 41 – Підключення до TDOA2 без введення паролю по протоколу SSH

Як бачимо, ім’я користувача змінилося з sdr@TDOA1 на sdr@TDOA2, що означає, що ми підключилися через SSH за допомогою RSA ключів і можемо керувати віддалено іншим мікрокомп’ютером. Для того, щоб відключитися від іншого мікрокомп’ютера, потрібно ввести в термінал команду:

exit

Ця команда дозволяє завершити поточний сеанс SSH і від'єднатися від віддаленого комп'ютера.

Аналогічні дії з копіювання відкритого RSA ключа потрібно проробити і з іншими, не основними, мікрокомп'ютерами Raspberry Pi. В нашому випадку це мікрокомп'ютер TDOA3. Якщо все зроблено правильно, то в результаті ми матимемо доступ до всіх мікрокомп'ютерів системи без необхідності вводу паролю, що дозволить створити bash-файли, які дозволять автоматично копіювати отримані файли з сигналами з усіх мікрокомп'ютерів на основний.

3.7 Розробка програмного забезпечення для одночасного прийому сигналів

Bash-файли або bash-скрипти – це текстові файли, що містять послідовність команд мови програмування Bash. Bash – це одна з популярних оболонок командного рядка в операційних системах на основі Linux та MacOS. Bash-скрипти дозволяють автоматично повторювати дії, такі як резервне копіювання, встановлення програмного забезпечення, оновлення системи тощо. Це підвищує ефективність роботи і заощаджує час.

Bash-скрипти можуть бути використані для створення власних командних інструментів, які розширюють функціональність операційної системи. Наприклад, можна створити скрипт для копіювання отриманих з приймачів файлів сигналу та часових міток на якийсь один мікрокомп'ютер, як в нашому випадку. Bash-скрипт створюється як звичайний текстовий файл з розширенням .sh, що досить просто для реалізації.

Для того, щоб створити bash-файл використаємо уже знайому програму WinSCP. Для цього підключимося до основного мікрокомп'ютера TDOA1 та створимо в директорії /home/sdr/ bash-файл з назвою TDOA та розширенням .sh, щоб в результаті отримати файл TDOA.sh. Відкриємо цей файл як текстовий документ і почнемо створювати скрипт.

На початку файлу завжди вказується інтерпретатор, який повинен виконати скрипт #!/bin/bash. У файлі містяться команди мови Bash, такі як змінні, умовні оператори, цикли чи функції. Команди виконуються

послідовно, зверху вниз, як у звичайному сценарії. #!/bin/bash – це шебанг директива, яка вказує операційній системі, який інтерпретатор необхідно використовувати для виконання файлу. Саме тому початок bash-файлу буде таким:

#script for operating the TDOA system

#!/bin/bash

Перший рядок це коментар для пояснення що за скрипт ми створюємо. Другий рядок – це шебанг-директива, яка вказує, що цей скрипт має виконуватися за допомогою bash-інтерпретатора, розташованого за шляхом /bin/bash. Наступний рядок не несе корисного навантаження, а слугує для розділення фрагментів:

echo "-----------------------------------"

Виводить на екран рядок, що складається з дефісів. Це можна використовуватися для візуального розділення виведення або для кращого форматування. Наступна частина коду буде перевіряти умову прийняття правильної кількості аргументів:

if [$# != 2]

then

echo "parameters missing, call: <scriptname> frequency1 num_samples_per_freq"

exit 1

fi

Це блок if-then-fi, який перевіряє, чи кількість аргументів, переданих при запуску скрипта, дорівнює 2. Якщо більш детально, то:

if [$# != 2] – перевіряє, чи кількість аргументів ($#) не дорівнює 2.

then – якщо умова if виконується (тобто аргументів не 2), виконується наступний блок.

echo "parameters missing, call: <scriptname> frequency1 num_samples_per_freq" – виводить на екран повідомлення про те, що бракує аргументів, і як правильно запускати скрипт.

exit 1 – завершує виконання скрипту з кодом помилки 1, вказуючи, що сталася помилка.

fi - завершує блок if-then. В мові програмування Bash блок умови завершується саме так.

Якщо кількість прийнятих аргументів саме 2, тоді продовжується виконання скрипту:

freq1=$1

num_samples=1.2e6

У цій частині значення прийнятих аргументів присвоюється змінним, а num_samples використовується для визначення кількості зразків даних, які потрібно скопіювати на основний Raspberry Pi. Змінна num_samples зберігає кількість таких зразків, яку потрібно передати для подальшої обробки в Matlab.

Далі виводиться інформація частоту, на якій приймається сигнал, та виводиться кількість зразків даних сигналу, які потрібно зберегти. Також виводиться роздільна полоса та інформація про дії, які будуть виконуватися далі:

echo "Specified parameters: reference frequency:" $freq1 ", samples_per_freq:" $num_samples

echo "----------"

echo "Login to PI Radios and capture data simultaneously"

Наступними рядками буде одночасний запис файлів з прийнятими сигналами на кожній Raspberry Pi, що використовується в системі, а також запис часових міток часу прийому сигналів:

rtl_sdr -f $freq1 -n $num_samples 1_test.dat && date +%s%9N > metadata_1.txt &

ssh sdr@192.168.88.233 "rtl_sdr -f $freq1 -n $num_samples 2_test.dat; date +%s%9N > metadata_2.txt" &

ssh sdr@192.168.88.237 "rtl_sdr -f $freq1 -n $num_samples 3_test.dat; date +%s%9N > metadata_3.txt"

Оскільки bash-файл виконується на одному Raspberry Pi, який ми обрали основним, то перший рядок запускає утиліту rtl_sdr на основному мікрокомп'ютері з встановленими параметрами, які ми передали на початку. Дані з основного мікрокомп'ютера будуть записані у файлі 1_test.dat. Після прийому сигналу одразу записується часова мітка фактичного часу, коли сигнал був записаний у файл metadata_1.txt. Далі бачимо оператор &\. Цей оператор дозволяє запускати команду в окремому процесі і одразу повертати управління до командного рядка, не чекаючи завершення виконання команди. Таким чином команди будуть виконані одночасно, що означає одночасний прийом і запис сигналу усіма приймачами. Наступні рядки запускають утиліту rtl_sdr з використанням SSH для підключення до інших Raspberry Pi. Так само, як і на основному мікрокомп'ютері, утиліта rtl_sdr записує файли сигналів на інших Raspberry Pi з назвами 2_test.dat та 3_test.dat, а також записуються відповідні часові мітки прийму сигналу в файли з назвами metadata_2.txt та metadata_3.txt.

Коли файли з відповідними параметрами одночасно записані, потрібно скопіювати ці файли на основний Raspberry Pi для подальшої обробки. Для цього в bash-файл потрібно ввести наступні рядки:

cp /home/sdr/1_test.dat /home/sdr/Matlab/1_test.dat

cp /home/sdr/metadata_1.txt /home/sdr/Matlab/metadata_1.txt

scp sdr@192.168.88.233:/home/sdr/2_test.dat /home/sdr/Matlab/2_test.dat

scp sdr@192.168.88.233:/home/sdr/metadata_2.txt /home/sdr/Matlab/metadata_2.txt

scp sdr@192.168.88.237:/home/sdr/3_test.dat /home/sdr/Matlab/3_test.dat

scp sdr@192.168.88.237:/home/sdr/metadata_3.txt /home/sdr/Matlab/metadata_3.txt

Перший рядок виводить на екран повідомлення, що буде виконане копіювання записаних файлів на основний мікрокомп'ютер. Копіювання буде здійснене в папку Matlab на основному мікрокомп'ютері. Більш детально:

cp /home/sdr/1_test.dat /home/sdr/Matlab/1_test.dat – копіює файл 1_test.dat, який був захоплений на локальному комп'ютері, в теку /home/sdr/Matlab/ на цьому ж комп'ютері.

cp /home/sdr/metadata_1.txt /home/sdr/Matlab/metadata_1.txt – копіює файл metadata_1.tx який був захоплений на локальному комп'ютері, в теку /home/sdr/Matlab/ на цьому ж комп'ютері.

Для копіювання файлів з віддалених машин, використовується команда scp:

scp sdr@192.168.88.233:/home/sdr/2_test.dat /home/sdr/Matlab/2_test.dat

scp sdr@192.168.88.233:/home/sdr/metadata_2.txt /home/sdr/Matlab/metadata_2.txt

scp sdr@192.168.88.237:/home/sdr/3_test.dat /home/sdr/Matlab/3_test.dat

scp sdr@192.168.88.237:/home/sdr/metadata_3.txt /home/sdr/Matlab/metadata_3.txt

Команда scp (Secure Copy) використовується для копіювання файлів 2_test.dat, 3_test.dat та metadata_2.txt, metadata_3.txt, які були захоплені з мікрокомп’ютерів під назвою TDOA2 та TDOA3 в теку /home/sdr/Matlab/ на основному комп'ютері TDOA1. Для цього використовується з'єднання SSH між мікрокомп’ютерами[27].

Після створення bash-файлу (додаток В), потрібно запустити його на основному Raspberry Pi, вписавши команду в термінал TDOA1 з відповідними параметрами сигналу, який ми хочемо прийняти. Для прикладу будемо приймати сигнал радіостанції на частоті 106.5 МГц. Команда для запуску bash-файлу виглядатиме наступним чином:

sh TDOA.sh 106.5e6 1e3

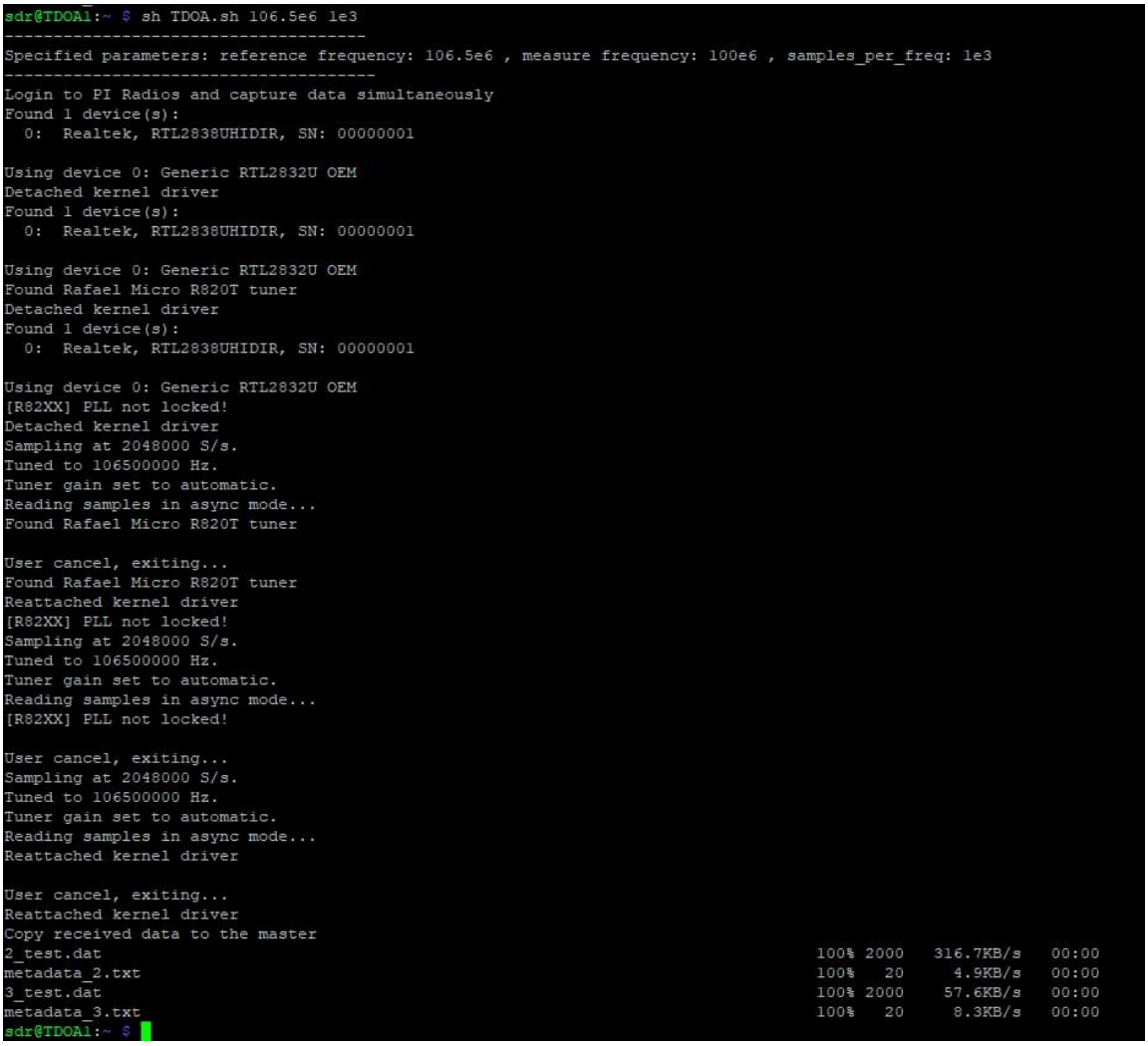

Рисунок 42 – Відображення в терміналі одночасного запису сигналу з отриманням часових міток та копіювання файлів на основний Raspberry Pi

В терміналі буде відображено модель приймача та частоту прийому сигналу, як на рис. 42. Результатом виконання bash-файлу є одночасний запис сигналів заданої частоти в файли та запис часових міток прийому сигналу, а також копіювання записаних файлів до директорії Matlab на основному мікрокомп’ютері.

3.8 Синхронізація часу за допомогою GPS модулю L76X GPS HAT

Різницево-далекомірна система, як різновид пасивної локації, визначає координати джерел випромінювання шляхом визначення різниці у часі прийому сигналу, що був прийнятий мінімум трьома приймачами. Знаючи точні координати приймачів та різницю в часі прийму цього сигналу, стає можливим визначення координат джерела випромінювання. Головним критерієм точного визначення координат джерела випромінювання є точність визначення часових затримок прийому сигналу. Для досягнення високої точності визначення місцезнаходження джерела сигналу, необхідно синхронізувати час на кожному Raspberry Pi. Синхронізація потрібна для досягнення однакового системного часу на мікрокомп’ютерах для створення часових міток прийому сигналу. Отже, точність визначення координат на пряму залежить від точності синхронізації.

Існує багато методів синхронізації часу, серед яких:

- NTPD – це демон, що дозволяє синхронізувати системний час між мікрокомп’ютерами, підключеними до однієї локальної мережі. Точність синхронізації від 1 до 100 мілісекунд.
- Systemd-timesyncd – це вбудований в систему Raspbian демон для синхронізації часу. Точність синхронізації від 10 до 100 мілісекунд.
- GPS синхронізація – це синхронізація, що відбувається з використанням глобальної системи позиціонування. Точність синхронізації від 20 до 100 наносекунд.
- PPS синхронізація – це синхронізація за допомогою високоточного сигналу PPS супутників. Точність синхронізації від 10 до 100 нанасекунд.

Усі наведені методи синхронізації дозволять створити часові мітки прийому сигналу для кожного приймача системи, але для TDOA методу визначення часових затримок необхідна синхронізація на рівні наносекунд.

Менша точність синхронізації негативно вплине на визначення місцеположення джерела сигналу[28].

Для ефективного визначення місцеположення джерел сигналу за допомогою TDOA методу необхідно використовувати методи, що забезпечують точність синхронізації на рівні наносекунд. Серед таких методів є синхронізація за сингалом GPS та сигналом PPS.

Синхронізація за допомогою PPS є найбільш точним методом, але цей метод потребує специфічного налаштування. Синхронізація за допомогою GPS має меншу точність, але в межах наносекунд, що задовольняє реалізацію TDOA системи. GPS має глобальне покриття. Тому використання GPS та PPS для синхронізації дозволяє отримати найбільш точну синхронізацію, але характеристики можуть бути відмінними від заявлених виробником зважаючи на погодні умови, пряму видимість супутників чи інші фактори, які знижують точність синхронізації з високоточними годинниками, що знаходяться на супутниках[29].

Для синхронізації часу за допомогою L76X GPS HAT потрібно підключити модуль до Raspberry Pi і перевірити чи правильно підключений модуль і чи отримує він данні з супутників за допомогою команди:

gpsmon

Ця команда запускає утиліту gpsmon, що дозволяє проводити моніторинг роботи модуля та показує чи приймає GPS модуль сигнали з супутників. Результат виконання команди зображено на рис. 43.

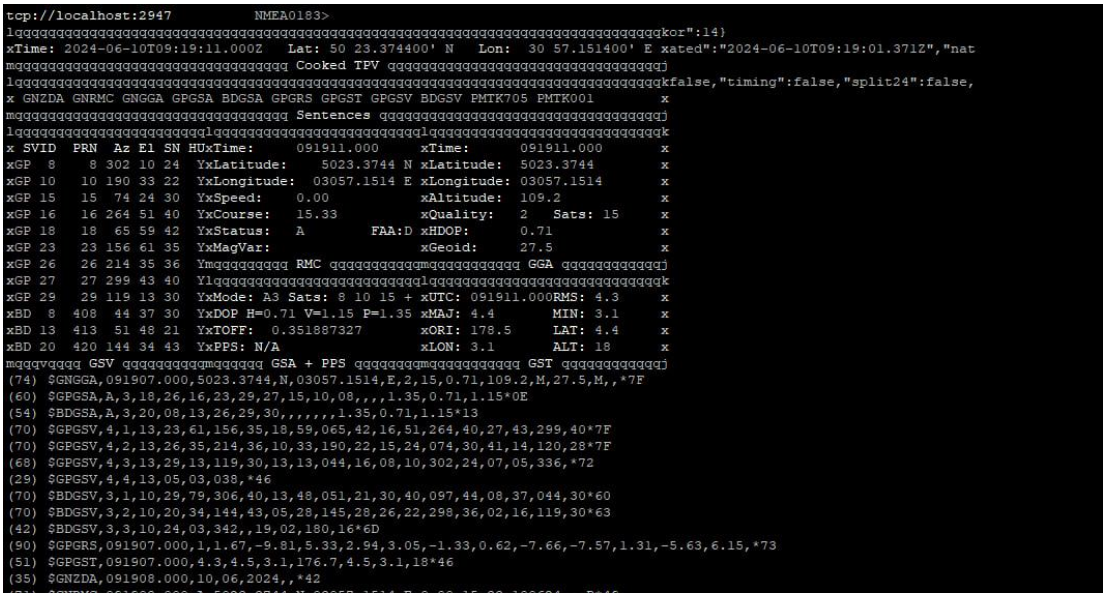

Рисунок 43 – Вигляд терміналу під час виконання команди gpsmon

Якщо сигнал з супутників приймається, наступним кроком буде встановлення chrony:

sudo apt-get install chrony

Chrony – це NTP клієнт для синхронізації системного часу Raspberry Pi з джерелами точного часу. Крім стандартних мережевих NTP-серверів, chrony має можливість синхронізації з GPS-приймачами, атомними годинниками та іншими джерелами точного часу.

Наступним кроком буде зупинка та відключення gpsd.socket для безпосередньої роботи клієнта chrony з gpsd:

sudo systemctl stop gpsd.socket

Ця команда зупиняє з’єднання з gpsd.socket, що використовується демоном gpsd за замовчуванням при підключенні пристрою GPS до системи. Відключення сокету означає відключення автоматичного запуску gpsd, що дає можливість запуску gpsd з потрібними параметрами та взаємодіяти з chrony безпосередньо для синхронізації часу. Для відключення автоматичного запуску gpsd.socket при завантаженні системи потрібно ввести команду:

sudo systemctl disable gpsd.socket

Далі потрібно запустити gpsd з прив’язкою до модулю L76X GPS HAT, який підключений через USB:

sudo gpsd /dev/ttyUSB0 -F /var/run/gpsd.sock

Після цього потрібно перезавантажити демон chrony для того, щоб демон почав отримувати данні про час безпосередньо від gpsd. Для цього потрібно ввести команду:

sudo systemctl restart chronyd

Після налаштування можна переглянути інформацію від джерел часу та перевірити чи було здійснено підключення chrony до gpsd. Для цього потрібно ввести в термінал команду:

chronyc sources

Результат виконання команди зображено на рис. 44.

```
sdr@TDOA3:~ $ chronyc sources
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
#- GPS                           0   4     0   794   +248ms[ +250ms] +/-  102ms
#- PPS                           0   4     0   475    +17us[  +20us] +/-   19us
^? host-80-254-30-78.sevsta>     0   8     0     -     +0ns[   +0ns] +/-    0ns
^* cache-a.dns.rx-name.net       2   6   376   186  +1016us[+1424us] +/- 7510us
^? ntp.time.in.ua                0   8     0     -     +0ns[   +0ns] +/-    0ns
^? 127-80-53-37.pool.ukrtel>     0   8     0     -     +0ns[   +0ns] +/-    0ns
^? ip-194-8-146-38.intelekt>     0   8     0     -     +0ns[   +0ns] +/-    0ns
^? ns2.infomir.com.ua            0   8     0     -     +0ns[   +0ns] +/-    0ns
^? 250-87-53-37.ip.ukrtel.n>     0   8     0     -     +0ns[   +0ns] +/-    0ns
^+ time.cloudflare.com           3   6   376   186  -2335us[-1926us] +/-   20ms
```

Рисунок 44 – Вивід інформації про джерела часу

Тепер можна отримати поточний час у форматі Unix. Такий формат часу показує час з початком відліку з 1 січня 1970 року 00:00:00 UTC. Початок відліку називається epoch.

Для виводу поточного часу у форматі Unix з врахуванням точності в наносекундах, потрібно ввести команду:

date +%s%9N

Ця команда дозволить перевірити синхронізацію часу на Raspberry Pi з часом GPS, а також дозволить в подальшому отримувати часові мітки часу прийому сигналу з кожного приймача. Для правильного визначення часових затримок потрібно повторити налаштування синхронізації для кожного Raspberry Pi системи[30].

Повний код для синхронізації:

gpsmon

sudo apt-get install chrony gpsd gpsd-clients

sudo systemctl stop gpsd.socket

sudo systemctl disable gpsd.socket

sudo gpsd /dev/ttyUSB0 -F /var/run/gpsd.sock

sudo systemctl restart chronyd

chronyc sources

date +%s%9N

Як результат, стане можливим виведення поточного часу, що синхронізований по GPS та PPS сигналах. Приклад виведення поточного часу показаний на рис. 45.

Рисунок 45 – Вивід поточного часу, що синхронізований з супутниками

3.9 Висновок до розділу 3

Було налаштовано програмне забезпечення, яке дозволяє прийняти радіосигнал, синхронізувати між собою Raspberry Pi з точністю до одиниць мікросекунд та визначити місцеположення приймачів системи. Розроблено програмне забезпечення, яке дозволяє керувати мікрокомп’ютерами Raspberry Pi, приймачами SDR та модулями GPS для одночасного прийому і передачі сигналів та метаданих про сигнал (час прийому) в середовище Matlab для визначення часових затримок між сигналами.

4. Експериментальне дослідження розробленого програмного забезпечення

Обране апаратне забезпечення та розроблене програмне забезпечення виконується за алгоритмом, зображеним на блок-схемі рис. 46.

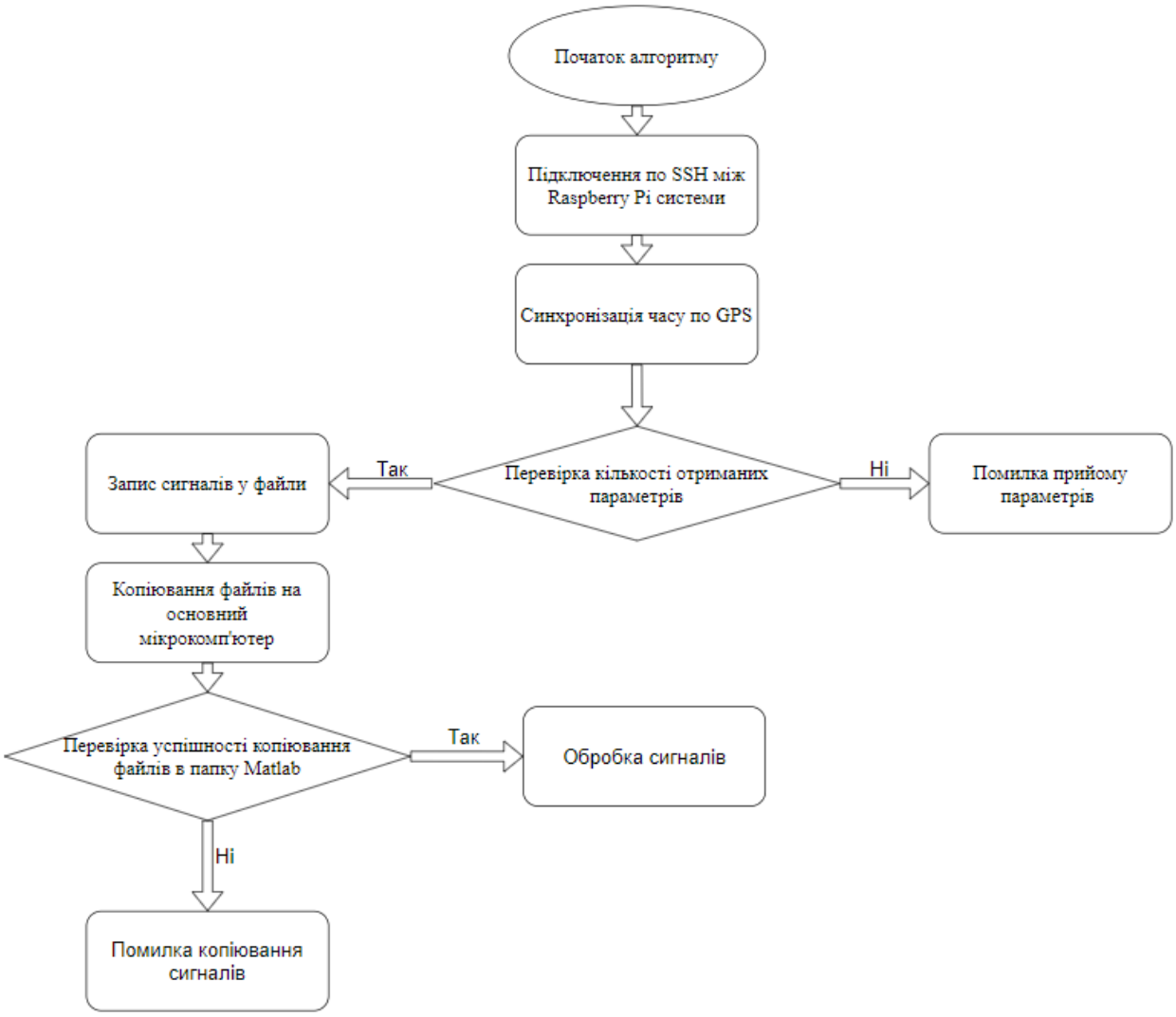

Рисунок. 46 – Блок-схема алгоритму визначення часових затримок

Отримавши за допомогою bash-скрипту часові мітки прийому сигналів та файли сигналів від кожного Raspberry Pi системи, потрібно врахувати час на обробку команд та їх передачу до всіх Raspberry Pi. Оскільки тестування апаратного та програмного забезпечення відбувалося в приміщенні, де всі приймачі та модулі GPS знаходилися в безпосередній близькості один до одного, то час прийому сигналів має бути приблизно однаковим. Часові мітки прийому сигналів створювалися для врахування затримок обробки команд

Raspberry Pi при обчисленні затримок часу прийому сигналу кожним приймачем.

З bash-скрипту видно, що хоч команди виконуються паралельно завдяки оператору &\ , проте час на виконання команд різний, оскільки основний Raspberry Pi виконує команди для прийому сигналу безпосередньо, а інші Raspberry Pi системи керуються за допомогою SSH з'єднання, що збільшує час для отримання сигналу:

rtl_sdr -f $freq1 -n $num_samples 1_test.dat && date +%s%9N > metadata_1.txt &

ssh sdr@192.168.88.233 "rtl_sdr -f $freq1 -n $num_samples 2_test.dat; date +%s%9N > metadata_2.txt" &

ssh sdr@192.168.88.237 "rtl_sdr -f $freq1 -n $num_samples 3_test.dat; date +%s%9N > metadata_3.txt"

Таким чином часові затримки між прийомом сигналу між TDOA1 та іншими мікрокомп'ютерами буде значно більшим ніж часові затримки між іншими мікрокомп'ютерами системи. Такі часові затримки виконання коду схематично зображені на рис. 47.

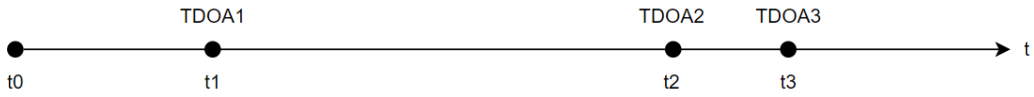

Рисунок 47 – Часова шкала затримок виконання команд

Для визначення часових затримок був використаний пакет прикладних програм для числового аналізу Matlab. Matlab має широку бібліотеку вбудованих функцій для обробки та відображення сигналів. Розроблений програмний код дозволяє визначити кореляцію між сигналами, отримати значення часових затримок між прийомом сигналів на основі їх кореляції та визначити часові затримки виконання команд кожним Raspberry Pi системи для визначення фактичних часових затримок між часом прийому сигналу [31].

Починається код програми з введення кількості приймачів системи для обробки прийнятих сигналів та часових міток прийому сигналів. При проведенні тестування було використано 3 приймачі, тому код виглядає наступним чином:

```
% Завантаження N сигналів
N = 3;
```

Оскільки сигнали збережені в бінарних файлах, то для зчитування та перетворення їх на комплексні числа з записом в змінні використовується функція read_complex_binary():

```
function data = read_complex_binary(filename)
    fid = fopen(filename, 'rb');
    raw = fread(fid, 'float32');
    fclose(fid);
    data = raw(1:2:end) + 1i*raw(2:2:end);
end
```

Ця функція відкриває бінарний файл та зчитує вміст файлу як одномірний масив чисел з плаваючою комою. Після зчитування файл закривається та створюється комплексний сигнал шляхом об’єднання уявної та дійсної частини.

Для зчитування часових міток використовується функція *load_metadata():*

```
function metadata = load_metadata(filename)
    fid = fopen(filename, 'r');
    metadata = textscan(fid, '%s', 'Delimiter', '\n');
    metadata = metadata{1};
    fclose(fid);
end
```

Функція відриває файл та зчитує вміст рядків, записуючи рядки в комірки metadata.

Наступним кроком потрібно виконати ініціалізацію комірки для зберігання сигналів та часових затримок :

```
signals = cell(1, N);
metadata = cell(1, N);
```

За допомогою циклів відбувається зчитування та запис даних у відповідні комірки:

```
for i = 1:N
   signals{i} = read_complex_binary(sprintf('%d_test.dat', i));
end
for i = 1:N
   metadata_filename = sprintf('metadata_%d.txt', i);
   metadata{i} = load_metadata(metadata_filename);
end
```

Далі потрібно ввести частоту дискретизації сигналів для визначення часових затримок. Під час виконання bash-файлу в терміналі відображається частота дискретизації сигналу, як на рисунку 17. Потрібно записати це значення частоти дискретизації в змінну sampling_rate:

```
sampling_rate = 2.048e6;
```

Далі для обчислення часових затримок виконання коду потрібно виконати ініціалізацію матрицю для збереження цих затримок:

```
mdelays = zeros(N,N);
```

Наступним кроком виконується цикл в циклі для запису в матрицю розрахунку різниці в часі виконання коду:

```
for i = 1:N
   for j = 1:N
      % Обчислити затримку
      mdelays(i,j) = abs(str2double(metadata{i}) - str2double(metadata{j}))*10^-9;
   end
end
```

Оскільки дані отримані в наносекундах, то відбувається перетворення в секунди, також результат береться по модулю, оскільки результатом розрахунків є часові значення.

Далі обчислюється матриця перехресних кореляцій. Для цього виконується ініціалізація комірок для зберігання кореляцій, затримок часу прийому сигналу та комірок для зберігання часових затримок з урахуванням часу на виконання коду:

```
crossCorrs = cell(N, N);
lags = cell(N, N);
delays = zeros(N, N);
True_delays = zeros(N, N);
```

Далі виконується цикл в циклі для обчислення та запису відповідних значень в комірки матриць:

```
for i = 1:N
   for j = 1:N
      [crossCorrs{i,j}, lags{i,j}] = xcorr(signals{i}, signals{j});
      % Знаходження піку кореляції
      [~, idx] = max(abs(crossCorrs{i,j}));
      % Обчислення затримок
      delay_in_samples = lags{i,j}(idx);
      delays(i,j) = abs(delay_in_samples / sampling_rate);
      % Обчислення затримок з урахуванням часу виконання коду
      True_delays(i,j) = delays(i,j) - mdelays(i,j);
   end
end
```

Функція xcorr відповідає формулі знаходження кореляції між сигналами:

$$Corr(k) = \sum_{n=0}^{N-1} s_1(n) \times s_2(n+k) \quad (1)$$

Далі відбувається виведення результатів:

fprintf('Матриця затримок часу (секунди):\n');

disp(delays);

fprintf('Матриця затримок часу з врахуванням затримок (секунди):\n');

disp(True_delays);

Під час тестування були отримані матриці затримок у часі, що відображені на рис. 46 та рис. 47, де індекси в матрицях відповідають індексам сингалів з приймачів.

```
Матриця затримок часу (секунди):
    0.0000    0.4829    0.4345
    0.4829    0.0000    0.0484
    0.4345    0.0484    0.0000
```

Рисунок 46 – Відображення матриці часових затримок між прийомом сигналів в секундах

На рис. 46 видно, що часові затримки становлять сотні мілісекунд між основним Raspberry Pi, в якому команди виконуються локально, та іншими Raspberry Pi, де команди надсилаються за допомогою SSH з'єднання. Затримки прийому сигналів між Raspberry Pi, що з'єднані по SSH, дещо нижчі, але все одно знаходяться на рівні десятків мілісекунд.

За допомогою визначення часових затримок на виконання команд та їх віднімання від загальних затримок, можна отримати значення часових затримок, що відповідатимуть безпосередньо різницям прийому сигналу. Результат зображений на рис. 47.

```
Матриця затримок часу з врахуванням затримок винаннаня коду (секунди):
   1.0e-05 *

    0.0000    0.1211    0.1182
    0.1211    0.0000    0.1211
    0.1182    0.1211    0.0000
```

Рисунок 47 – Відображення матриці часових затримок безпосереднього прийому сигналу

4.1 Висновки до розділу 4

Проведено аналіз часової різниці запису сигналу на різні SDR приймачі. Отримані результати виконання команд та визначено часові затримки прийому сигналів. Різниця часу прийому сигналу знаходиться в межах одиниць мікросекунд.

ВИСНОВКИ

1. Проведено аналіз систем визначення місцеположення джерел радіовипромінювання, їх переваги та недоліки.

2. На основі аналізу обрано апаратне забезпечення для побудови різницево-далекомірної системи.

3. Розроблено програмне забезпечення, яке дозволяє прийняти радіосигнал, синхронізувати між собою Raspberry Pi та визначити місцеположення приймачів.

4. Розроблено програмне забезпечення для визначення часових затримок сигналу між приймачами.

5. Проведено експериментальне дослідження, що показало працездатність апаратного та програмного забезпечення. Отримані часові затримки становлять одиниці мікросекунд.

Загалом, обране апаратне та розроблене програмне забезпечення дозволило виміряти часові затримки між прийнятими сигналами, що є основою для реалізації різницево-далекомірної системи. Подальше вдосконалення апаратного забезпечення дасть можливість збільшити точність прийому сингалів та синхронізації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Classification of Uncrewed Aerial Vehicles [Електронний ресурс] – Режим доступу до ресурсу: https://encyclopedia.pub/entry/43656

2. Системи виявлення дронів і протидронні системи [Електронний ресурс] – Режим доступу до ресурсу: https://www.bezpeka-shop.com/ua/blog/obzor/sistemy-obnaruzheniya-dronov-i-protivodronnye-sistemy/

3. DroneTracker [Електронний ресурс] – Режим доступу до ресурсу: https://www.dronov.net.ua/dedrone.php

4. DroneShield DroneSentinel [Електронний ресурс] – Режим доступу до ресурсу: https://drone-detection-system.com/sensor-types-overview/anti-drone-jammer/?gad_source=1&gclid=Cj0KCQjw0ruyBhDuARIsANSZ3wrvfLrVA2AfNg3hCisEXGDieXb7XPdzl-yYIzBgr9-lFdrRRcqPZloaAm7wEALw_wcB

5. DEDRONE DRONETRACKER [Електронний ресурс] – Режим доступу до ресурсу: https://www.dronov.net.ua/dedrone.php

6. Система CerbAir [Електронний ресурс] – Режим доступу до ресурсу: https://www.getac.com/ru/success-stories/cerbair-революция-вобласти-безопасности/

7. Радіоприймачі [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Радіоприймач#:~:text=Радіоприймáч%20—%20пристрій%2C%20призначений%20для%20приймання,якому%20вона%20може%20бути%20використана.

8. Software Defined Radio [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Software_Defined_Radio

9. Трансивер HackRF One SDR [Електронний ресурс] – Режим доступу до ресурсу: https://arduino.ua/ru/prod5198-hackrf-one

10. Airspy R2 SDR приймач [Електронний ресурс] – Режим доступу до ресурсу: https://radioscan.com.ua/ua/p2071721018-airspy-sdr-priyomnik.html

11. ADALM-PLUTO [Електронний ресурс] – Режим доступу до ресурсу: https://evo.net.ua/ru/adalm-pluto/

12. LimeSDR [Електронний ресурс] – Режим доступу до ресурсу: https://radioscan.com.ua/ua/p2172857726-limesdr-transiver-100.html

13. RTL-SDR [Електронний ресурс] – Режим доступу до ресурсу: https://sdr.in.ua/product/rtl-sdr-v4/

14. Мікрокомп'ютер [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Мікрокомп%27ютер#:~:text=Мікрокомп%27ютер%20—%20термін%20для%20позначення,10-х%20років%20XXI%20століття.

15. Мікрокомп'ютер Raspberry Pi 3 Model B+ [Електронний ресурс] – Режим доступу до ресурсу: https://evo.net.ua/ru/raspberry-pi-3-model-b/

16. Orange Pi 5 8G [Електронний ресурс] – Режим доступу до ресурсу: https://arduino.ua/ru/prod5627-orange-pi-5-8g

17. Плата розширення Raspberry Pi LC29H(AA) GPS HAT (25278) [Електронний ресурс] – Режим доступу до ресурсу: https://evo.net.ua/ru/plata-rozshyrennia-raspberry-pi-lc29haa-gps-hat-25278/

18. ПЛАТА РОЗШИРЕННЯ GPS NEO-6M ДЛЯ RPI [Електронний ресурс] – Режим доступу до ресурсу: https://miniboard.com.ua/platy-rasshireniya/373-plata-rasshireniya-gps-neo-6m-dlya-rpi.html

19. L76X GPS HAT [Електронний ресурс] – Режим доступу до ресурсу: https://www.waveshare.com/wiki/L76X_GPS_HAT#Working_with_Sunrise_X3_PI

20. Raspberry Pi Software [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Raspberry_Pi#Software

21. Recording the operating system on the Raspberry Pi [Електронний ресурс] – Режим доступу до ресурсу: https://www.raspberrypi.com/documentation/computers/getting-started.html

22. Activation of SSH in the Raspberry Pi terminal [Електронний ресурс] – Режим доступу до ресурсу: https://www.onlogic.com/blog/how-to-ssh-into-raspberry-pi/

23. Installing drivers for RTL-SDR [Електронний ресурс] – Режим доступу до ресурсу: https://osmocom.org/projects/rtl-sdr/wiki/Rtl-sdr

24. Documentation About WinSCP [Електронний ресурс] – Режим доступу до ресурсу: https://winscp.net/eng/docs/introduction

25. Configuring Wi-Fi on Raspberry Pi 4 [Електронний ресурс] – Режим доступу до ресурсу: https://binaryupdates.com/how-to-configure-wifi-on-raspberry-pi-4/

26. Connection to SSH Without a Password [Електронний ресурс] – Режим доступу до ресурсу: https://builtin.com/articles/ssh-without-password

27. Bash Scripting [Електронний ресурс] – Режим доступу до ресурсу: https://www.freecodecamp.org/news/bash-scripting-tutorial-linux-shell-script-and-command-line-for-beginners/#:~:text=A%20bash%20script%20is%20a,process%20using%20the%20command%20line.

28. Ways to synchronize time between Raspberry Pi [Електронний ресурс] – Режим доступу до ресурсу: https://forums.raspberrypi.com/viewtopic.php?t=151605

29. Using a GPS module to set the correct time for the Raspberry PI [Електронний ресурс] – Режим доступу до ресурсу: https://www.haraldkreuzer.net/en/news/using-gps-module-set-correct-time-raspberry-pi-3-a-plus-without-network

30. Time synchronization using GPS and GPS [Електронний ресурс] – Режим доступу до ресурсу: https://linlog.blogspot.com/2009/07/synchronizing-ntp-server-to-gpspps.html

31. Matlab documentation [Електронний ресурс] – Режим доступу до ресурсу: https://ch.mathworks.com/help/referencelist.html?type=function

ДОДАТОК А. КОД НАЛАШТУВАННЯ SDR ПРИЙМАЧА

```
sudo apt update
sudo apt-get install git
sudo apt-get install cmake
sudo apt-get install build-essential

sudo apt install libusb-1.0-0-dev
git clone https://gitea.osmocom.org/sdr/rtl-sdr.git
cd rtl-sdr
mkdir build
cd build
cmake ../ -DINSTALL_UDEV_RULES=ON -DDETACH_KERNEL_DRIVER=ON
make
sudo make install
sudo ldconfig
sudo reboot
rtl_test
```

ДОДАТОК Б. КОД НАЛАШТУВАННЯ L76X GPS HAT

```
sudo apt-get update
sudo pip install RPi.GPIO
sudo apt-get install python3-serial
sudo apt-get install gpsd gpsd-clients python3-gps

sudo apt-get install minicom
sudo minicom -D /dev/ttyUSB0 -b 9600

wget  https://files.waveshare.com/upload/f/f4/L76X_GPS_HAT_Code.zip
unzip L76X_GPS_HAT_Code.zip
sudo chmod 777 -R L76X_GPS_HAT_Code
cd L76X_GPS_HAT_Code/RaspberryPi

cd python
sudo pip3 install pynmeagps
sudo pip3 install gps3
sudo python3 main.py
```

ДОДАТОК В. КОД BASH-СКРИПТА

```
#script for operating the TDOA system
#!/bin/bash
echo "-------------------------------------"
if [ $# != 2 ]
  then
   echo "parameters missing, call: <scriptname> frequency1 num_samples_per_freq"
    exit 1
fi
freq1=$1
num_samples=$2
echo "Specified parameters: reference frequency:" $freq1 ", samples_per_freq:" $num_samples
echo "Login to PI Radios and capture data simultaneously"
rtl_sdr -f $freq1 -n $num_samples 1_test.dat && date +%s%9N > metadata_1.txt &\
ssh sdr@192.168.88.233 "rtl_sdr -f $freq1 -n $num_samples 2_test.dat; date +%s%9N > metadata_2.txt"  &\
ssh sdr@192.168.88.237 "rtl_sdr -f $freq1 -n $num_samples 3_test.dat; date +%s%9N > metadata_3.txt"
echo "Copy received data to the master"
cp /home/sdr/1_test.dat /home/sdr/Matlab/1_test.dat
cp /home/sdr/metadata_1.txt /home/sdr/Matlab/metadata_1.txt
scp sdr@192.168.88.233:/home/sdr/2_test.dat /home/sdr/Matlab/2_test.dat
scp sdr@192.168.88.233:/home/sdr/metadata_2.txt /home/sdr/Matlab/metadata_2.txt
scp sdr@192.168.88.237:/home/sdr/3_test.dat /home/sdr/Matlab/3_test.dat
scp sdr@192.168.88.237:/home/sdr/metadata_3.txt /home/sdr/Matlab/metadata_3.txt
```